SESSION

REAL-WORLD DATA MINING APPLICATIONS, CHALLENGES, AND PERSPECTIVES

Chair(s)

Mahmoud Abou-Nasr
Robert Stahlbock
Gary M. Weiss

ISBN: 1-60132-431-6, CSREA Press ©

ISBN: 1-60132-431-6, CSREA Press ©

Clustering and Prediction of Solar Radiation Daily Patterns

G. Nunnari[1], **and S. Nunnari**[1]
[1]Dipartimento di Ingegneria Elettrica, Elettronica e Informatica, Universitá di Catania, Catania, Italy

Abstract— *This paper addresses the problem of clustering daily patterns of global horizontal solar radiation by using a feature-based approach. A pair of features, referred to as S_r and H_r, representing a measure of the normalized daily solar energy and of the energy fluctuations, respectively, is introduced. Clustering allows to perform some useful statistics at daily scale such as estimating the class weight and persistence. Furthermore, the problem of one-day ahead prediction of the class is addressed by using both hidden Markov models (HMM) and Non-linear Autoregressive (NAR) models. Performances are then assessed in terms of True Positive Rate (TPR) and True Negative Rate (TNR).*

Keywords: Solar Radiation, HMM models, NAR models, Clustering time series, Prediction.

1. Introduction

Solar radiation is a quite irregular kind of time series, due several complex processes such as the clouds cover features, the atmospheric transmittance, the sky turbidity and the pollution level. One problem, dealing with solar radiation time series is that they are scarcely auto-correlated. In particular, at hourly scale the correlation time is about 5 lags, while at daily scale the correlation time is 1 lag (see for instance [1]). Thus while there is some chance to forecast hourly average solar radiation at short term by using auto-regressive models, there are very limited possibilities of forecasting one-day ahead the average value of solar radiation. For this reason, clustering techniques may provide useful tools to get some statistical information at daily scale. Previous work concerning the clustering of solar daily patterns was carried out by [2], based on daily distributions of the clearness index. Classification of solar radiation patterns into three classes, referred to as overcast, partly cloudy and sunny, was proposed by [3], based on the use of 10 min sampling data. Classification into to four classes based on 5m sampling data was proposed by [4].
In this paper we propose a strategy belonging to the class of methods working on features extracted from hourly average samples, which are more widely available from public data base with respect to 10m or 5m average time series. Indeed, different sampling rate requires different algorithms for feature extracting, mainly for measuring the degree of fluctuation of solar radiation at daily scale. The paper is organized as follows: a short description of the considered data set is provided in section 2, while description of features extracted from solar radiation daily patterns is given in section 3. The adopted clustering strategy is described in section 4, while applications, devoted to perform some statistical insights from the time series of classes, is given in section 5. One-day ahead class prediction is addressed in section 6 and, finally, conclusions are traced in section 7.

Table 1: Geographic coordinates of the 12 solar radiation recording stations belonging to the considered data set.

stationID	Lat	Lon	$Elev$	UTC
690140	33.667	-117.733	116	-8
690150	34.3	116.167	626	-8
722020	25.817	-80.3	11	-5
722350	32.317	-90.083	94	-6
722636	36.017	-102.55	1216	-6
723647	35.133	-106.783	1779	-7
724776	38.58	-109.54	1000	-7
725033	40.783	-73.967	40	-5
725090	42.367	-71.017	6	-5
726055	43.083	-70.817	31	-5
726130	44.267	-71.3	1910	-5
726590	45.45	-98.417	398	-6

2. The considered solar radiation data set

The data set considered in this paper consists of hourly average time series recorded at twelve stations stored in the USA National Solar Radiation Database managed by the NREL (National Renewable Energy Laboratory). Data of this database was recorded from 1999 to 2005 and can be freely download from ftp://ftp.ncdc.noaa.gov/pub/data/nsrdb-solar/. The twelve stations were selected based on two criteria: the quality of time series and the need to ensure the necessary diversification of meteo-climatic conditions. The twelve selected stations are listed in Table (1). More detailed information about these and others recording stations of the National Solar Radiation Database can be found in [5].

3. Two features of solar radiation

In order to choose a limited number of representative features of the solar radiation time series, it is quite natural to choose a pair that can represent the quantity of solar radiation recorded during a day and the level of fluctuation. Evidently, while the first feature is directly connected with the quantity of electrical energy per day that is expected to

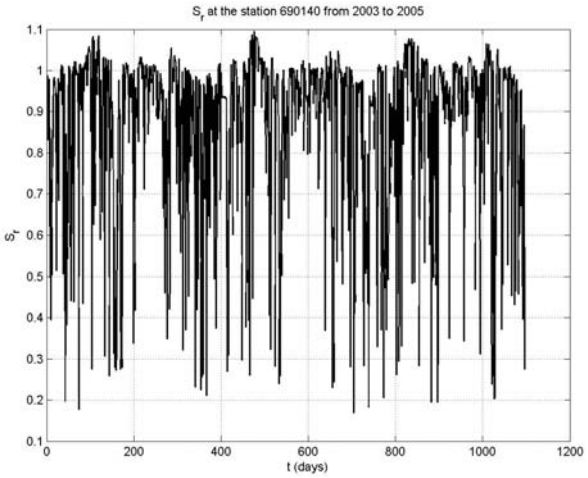


Fig. 1: Daily values of the S_r feature computed for the station ID690140 from 2003 to 2005.

produce, the second gives a measure of its intermittency. The formal definition of the proposed features is given below.

3.1 The solar radiation ratio S_r

The S_r feature is formally represented by expression (1)

$$S_r(t) = \frac{S_{pat}(t)}{S_{csk}(t)} \tag{1}$$

where $S_{pat}(t)$ and $S_{csk}(t)$ represent the area under the true and the global horizontal solar irradiation daily patterns in clear sky conditions, respectively, and t is the time expressed in days. The $S_{csk}(t)$ can be computed referring to one of several existing clear sky models, such as the Ineichen and Perez model [6], [7]. The Matlab code to implement such a model is part of the SNL_PVLib Toolbox, developed in the framework of the Sandia National Labs PV Modeling Collaborative (PVMC) platform. Of course the S_r feature is always positive but can be greater or less than 1: in a day featured by favorable weather conditions (e.g. absence of cloud cover and good atmospheric transmittance) S_r can be slightly greater than 1; conversely under thick cloud cover and adverse propagation conditions it may be significantly less than 1. As an example, the behavior of $S_r(t)$ computed for the station ID690140 from 2003 to 2005 is reported in Figure (1).

3.2 The Hurst exponent ratio

The Hurst exponent ratio H_r is formally represented by expression (2)

$$H_r(t) = \frac{H_{pat}(t)}{H_{csk}(t)} \tag{2}$$

where $H_{pat}(t)$ and $H_{csk}(t)$ are the Hurst exponent of the true and clear sky solar radiation patterns, respectively, at the generic day t. The rationale for this definition is that while smooth curves, such as solar radiation in clear sky conditions, gives H_r close or slightly greater than 1, the presence of fluctuations generates H_r less than 1. For the purposes of this paper the Hurst exponent was computed by using the Geweke-Porter-Hudak (GPH) algorithm, first described by [8]. The reason for using this algorithm, instead of the more popular R/S and DFA algorithms, is that it is considered more appropriate when patterns, as in this paper, are represented by a small number of samples. Indeed, the GPH algorithm is based on the slope of the spectral density function. An example of H_r index, computed for the station ID690140 during three years, is shown in Figure (2).

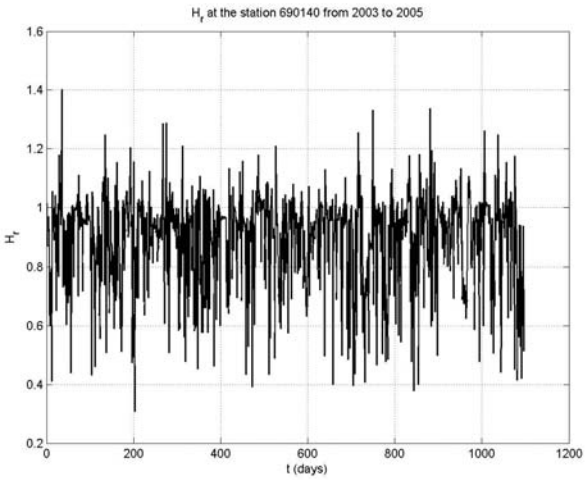


Fig. 2: H_r daily values computed for the station ID690140 from 2003 to 2005.

3.3 Some properties of the S_r and H_r features

The extremely scattered behavior of these features can be objectively evaluated by computing the mutual information of the individual S_r and H_r, as shown in Figure (3). Indeed, the mutual information suddenly decays after just 1 lag. This result implies that one-day ahead prediction of the class is extremely difficult by using auto-regressive models. Therefore, clustering approaches can be useful to extract, at least, some statistical information at daily scale. It is worth noting that S_r and H_r are not really independent features, as shown by the cross-correlation function reported in Figure (4). In more detail, they are positive-correlated, thus meaning that high H_r correspond to high S_r, as mentioned in section 3.2. The role of S_r and H_r to classify solar radiation daily patterns is discussed in the next section.

4. Clustering solar radiation features

The general goal of clustering is to identify possible structures in an unlabeled data set, in such a way that data is objectively organized in homogeneous groups. The heart of

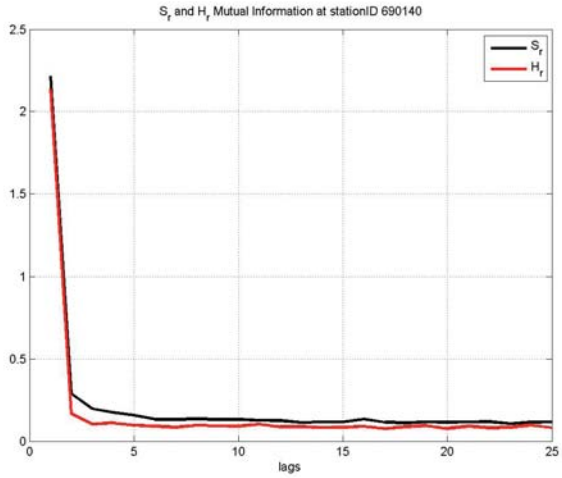


Fig. 3: Mutual information of Sr and H_r computed for the station ID690140.

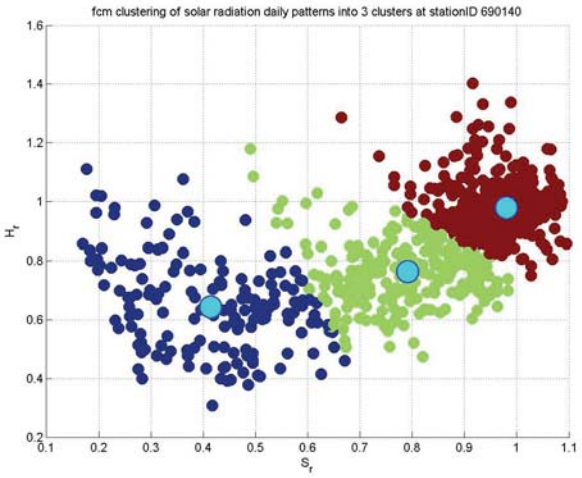


Fig. 5: Pattern distribution and cluster centers computed for the station ID690140.

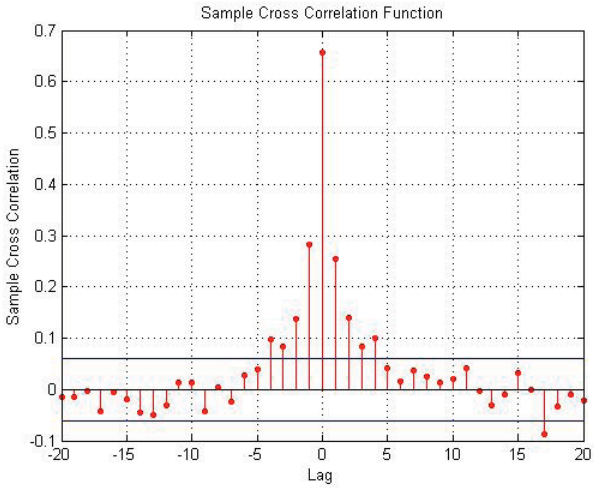


Fig. 4: Cross-correlation between Sr and H_r computed for the station ID690140.

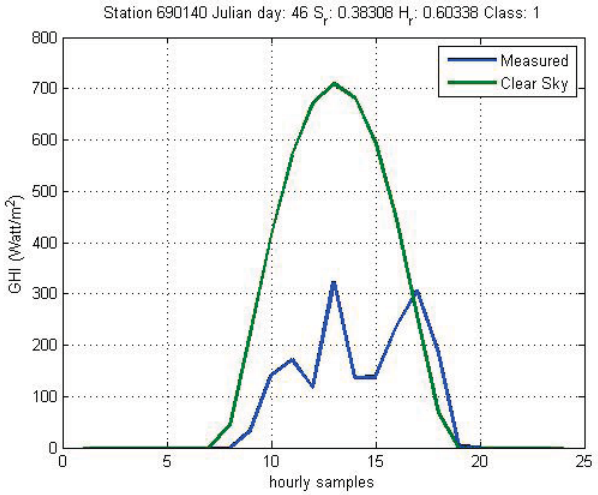


Fig. 6: Example of class C_1 pattern at the station ID690140.

any clustering approach, regardless of whether the problem is to classify static data or time series, is a clustering algorithm. The most important existing algorithms are usually grouped into four groups: exclusive clustering, overlapping clustering hierarchical clustering and probabilistic clustering. For the purposes of this work, since it is not realistic to perform several clustering levels, hierarchical clustering is not appropriate. In this paper, we have considered the fuzzy-c means (*fcm*) algorithm, which is simple to implement, allows overlapping clustering and generate only one level of clusters. Results of clustering into 3 classes the features computed for the station ID690140 is shown in Figure (5). As it is possible to see the *fcm* algorithm essentially distributes the cluster centers for increasing values of S_r which thus play the role of primary feature. Furthermore, as already observed, patterns characterized by high S_r are also featured by high values of H_r.

Representative patterns, in a 3-class framework, are shown in Figures (6), (7) and (8), respectively. In order to assess the consistency of clustering by using the *fcm* algorithm, we have computed the silhouette, which for the station ID690140 looks as in Figure (9). The silhouette $S(i)$ is a measure, ranging from -1 to 1, of how well the i_{th} pattern lies within its cluster: $S(i)$ close to 1 means that the corresponding pattern is appropriately clustered; on the contrary, if $S(i)$ is close to -1, then the i_{th} pattern would be more appropriate if it was clustered in its neighboring cluster; finally $S(i)$ near to zero means that the pattern is on the border of two natural clusters. As it is possible

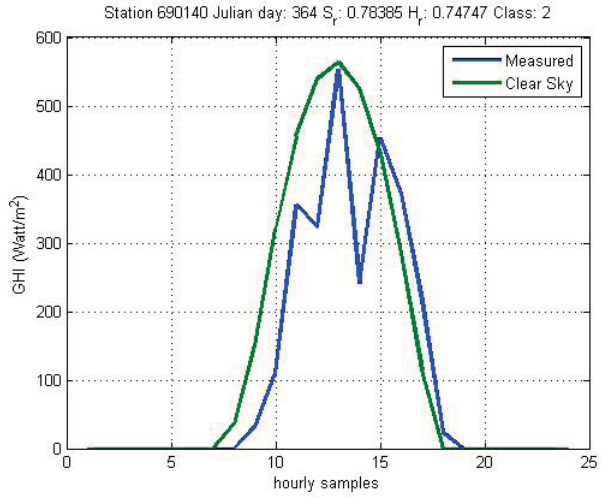


Fig. 7: Example of class C_2 pattern at the station ID690140.

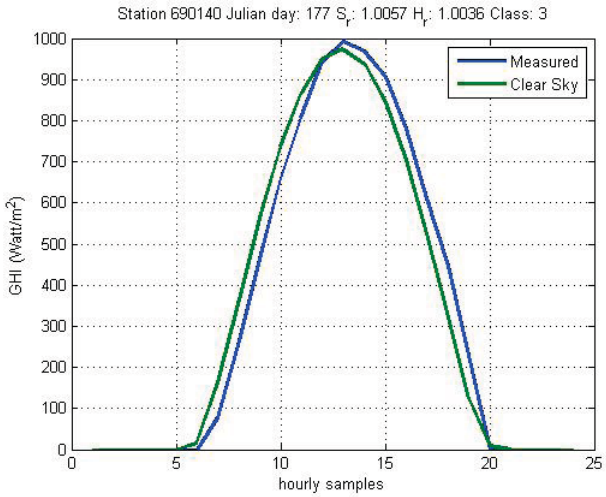


Fig. 8: Example of class C_3 pattern at the station ID690140.

to appreciate from Figure (9), only a limited number of events of classes C_1 and C_2 are characterized by negative silhouette, thus assessing the goodness of the clustering.

It is to be stressed that the coordinates of the cluster centers are depending on the recording site, as shown in Figure (10), where the centers computed at the 12 considered stations, within a 3-class framework, are reported.

5. Some applications

Once classes have been attributed to daily wind speed time series, some useful statistics can be carried out, such as, for instance, computing the weight of a class and the persistence of patterns in a class.

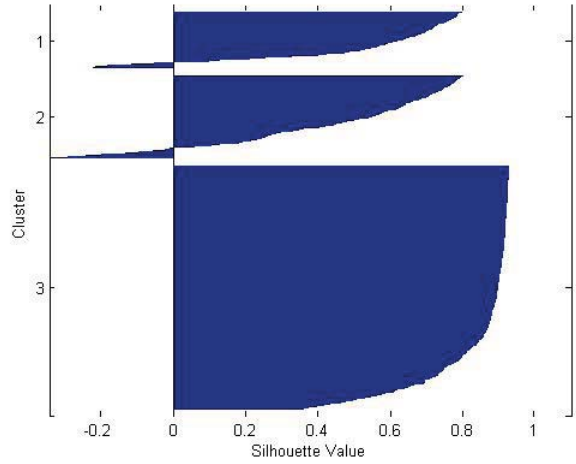


Fig. 9: Silhouette computed clustering patterns of the station ID690140 into 3 classes by the *fcm* algorithm.

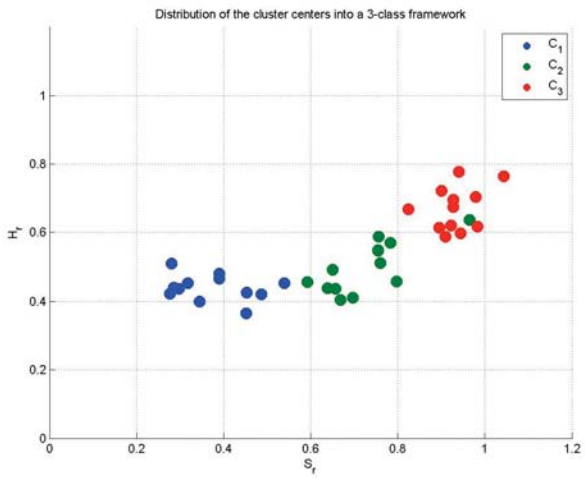


Fig. 10: Distribution of the cluster centers at 12 stations into a 3-class framework.

5.1 Weight of a class

The weight W_i of a class C_i is formally defined as in (3)

$$W_i\% = \frac{n_i}{\sum_{i=1}^{n_c} n_i} 100 \tag{3}$$

where n_i and n_c are the number of patterns in class C_i and the number of classes, respectively. The weights computed for each station of the considered data set are reported in Table (2), As it is possible to see, the weights are heavily affected by the different meteo-climatic conditions in which operates each individual recording station. In particular, Table (2) shows that class C_3 exhibits on average the highest weight.

Table 2: Weight in percent of the four classes at 12 stations.

stationID	$W_1\%$	$W_2\%$	$W_3\%$
690140	14.69	21.53	63.78
690150	11.13	30.38	58.49
722020	23.18	34.03	42.79
722350	21.72	24.45	53.83
722636	15.51	24.73	59.76
723647	13.78	24.27	61.95
724776	18.16	29.47	52.37
725033	27.46	30.57	41.97
725090	26.55	27.01	46.44
726055	27.01	27.83	45.16
726130	31.20	37.04	31.75
726590	24.73	29.20	46.08
Average W	**21.26**	**28.38**	**50.36**

5.2 Estimating the persistence

A useful statistic is that of estimating the persistence, defined as the number of episodes in a year in which a daily pattern persists in the same class for at least p consecutive days. An example of this kind of statistic is given in Figure (11). The Figure refers to the persistence of patterns at the

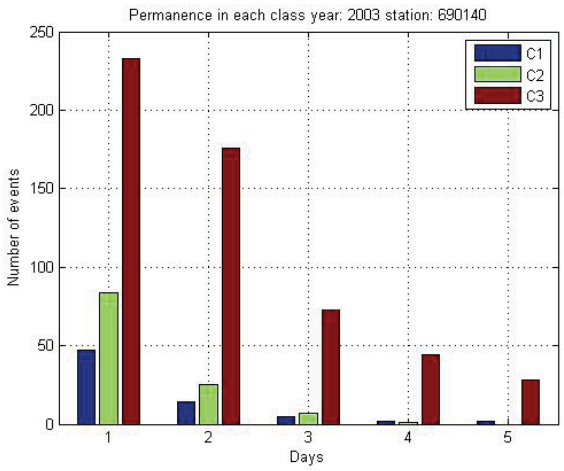


Fig. 11: Persistence in the same class at the station ID690140 during 2003.

station ID690140 during 2003 and show, for instance, that 176 events lasted in class C_3 at least two days, 73 at least 3 days, 44 of at least 4 days and 28 at least 5 days. Instead, for patterns of class C_2, only 25 lasted at least 2 days, 7 at least 3 days and 1 at least 4 days. This kind of information could be useful to a solar plant manager in the absence of reliable predictions at daily scale.

6. Predicting one-day ahead the class

In this section we report results concerning some attempts to predict one-day ahead the time series of solar radiation class, obtained as described in the previous section 4. Of course, the prediction problem is as difficult as larger is the number of considered classes. Here we report results concerning prediction in 2 and 3 class frameworks, since as explained in section 6.4, at the present stage of this work, results are not reliable for larger number of classes. As concerning the prediction approaches we have considered HMM and NAR models.

6.1 Predicting by using HMM models

A HMM [15] is a modeling approach in which we observe a sequence of emissions, but we do not know the sequence of states the model went through to generate the emissions. Thus, in general a HMM model is characterized by two matrices, referred to as the transition matrix and the emission matrices, respectively. Prediction by using this kind of models requires that the emission matrix must be transformed into the most probable state path by using one of several available algorithms, such as the popular Viterbi algorithm [16].

6.2 Predicting by using NAR model

The NAR-based model consists of two main steps. In the first step, one-day-ahead prediction of $\hat{S}_r(t+1)$ and $\hat{H}_r(t+1)$) of the two features $S_r(t)$ and $H_r(t)$ are performed by using independent models of the form (4).

$$y(t+1) = f(y(t), y(t-1), ..., y(t-d+1)) \qquad (4)$$

In expression (4) d (dimension) is the number of considered past values and f a non-linear unknown map, here identified by using a neural network approach. In the second step, the predicted class $c(t+1)$ is obtained by using a neural network classifier, previously trained to assign a class to a predicted pair of features ($\hat{S}_r(t+1)$,$\hat{H}_r(t+1)$)).

6.3 Performance indices

In order to objectively asses to what extent a predicted time series of classes is close to the true one it is possible to use several performance indices. In particular, in this paper we have considered the TPR (True Predicted Rate) and the TNR (True Negative Rate), defined as follows:

$$TPR(i) = \frac{TP(i)}{TP(i)+FN(i)} \qquad (5)$$

$$TNR(i) = \frac{TN(i)}{TN(i)+FP(i)} \qquad (6)$$

where $TP(i)$ and $FN(i)$ is the number of true positive and false positive patterns, respectively, attributed by the model to the class C_i and i is the class index. The sum $P(i) = TP(i) + FN(i)$ is, of course the total number of patterns attributed by the model to the class C_i. Similarly, $TN(i)$ is the number of patterns which are correctly identified as not belonging to the class C_i and $FP(i)$ is the number of false positives attributed by the model to the class C_i. The sum $N(i) = TN(i) + FP(i)$ is the total number of patterns recognized by the model as not belonging to the

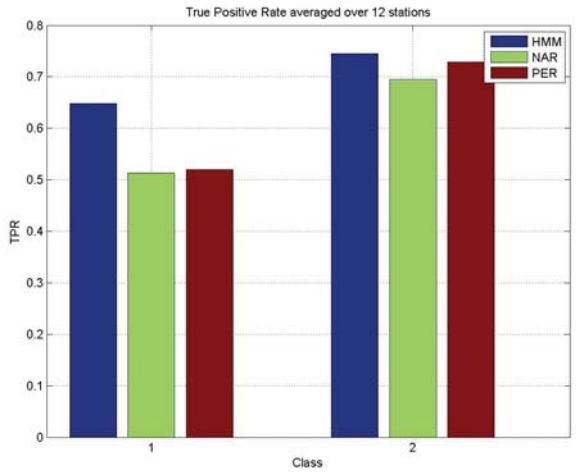


Fig. 12: TPR for the 2-class framework averaged over twelve stations.

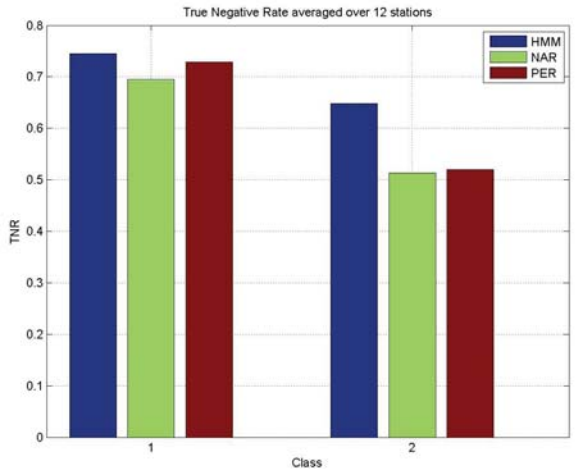


Fig. 13: TNR for the 2-class framework averaged over twelve stations.

class C_i. Clearly, a good predictor would be characterized by values of TPR and TNR both close to 1. Performances of the HMM and NAR models were also compared with the simple persistent model (7), often considered as a low reference model.

$$c(t+1) = c(t) \tag{7}$$

6.4 Numerical results

Results described in this section was obtained by using for each considered station hourly average solar radiation time series, recorded from 2003 to 2005. For each station, two years of data (2003 and 2004) was considered to identify the HMM and NAR prediction models, while the remaining year 2005 was considered to test the models. In order to generalize the results, the performance indices shown in this section were averaged over the whole set of considered stations.

6.4.1 Prediction into a 2-class framework

We start the description with the simplest case, i.e. the prediction in a 2-class framework. The TPR and TNR, averaged over twelve stations are reported in Figure (12) and (13), respectively. It is possible to see that the HMM prediction model outperform, both the NAR and the persistent models, exhibiting an average TPR of about 0.65 and 0.75 for the classes C_1 and C_2. Similarly, the TNR is about 0.75 and 0.65 for the classes C_1 and C_2, respectively. A detail of the TPR and TNR of the HMM model obtained for each individual station is reported in Figure (14) and (15), respectively, which show that the model performances, are significantly affected by the recording site. In this Figure, different colors refer to different stations; the order of the station is the same as in the first column of Table (2). In the background of the Figure the TPR averaged over the twelve stations is given.

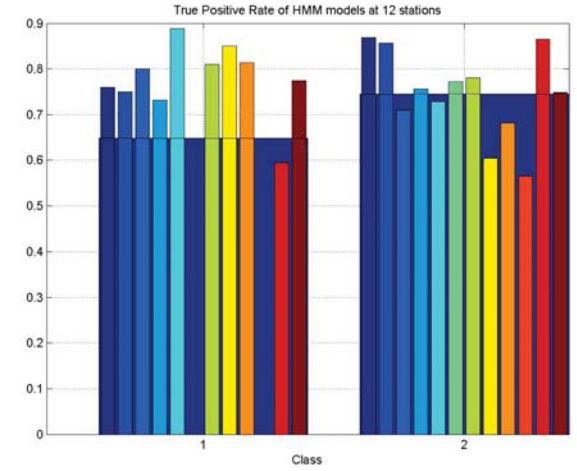


Fig. 14: TPR of the HMM model for each of the 12 station, in the 2-class framework.

6.4.2 Prediction into a 3-class framework

As concerning the models performances into a 3-class framework, the average TPR and TNR are reported in Figure (16) and (17), respectively. In particular, Figure (16) shows that despite the considered models are capable of predicting an acceptable proportion of patterns belonging to the class C_3, they are not as good to correctly predict patterns belonging to the classes C_1 and $C2$, since the corresponding TPR is below 0.5. That notwithstanding, Figure (17) shows that all models have a good ability to correctly recognize patterns that do not belong to a particular class. Furthermore, for the 3-class framework, there is not a clear prevalence of a model with respect to the other.

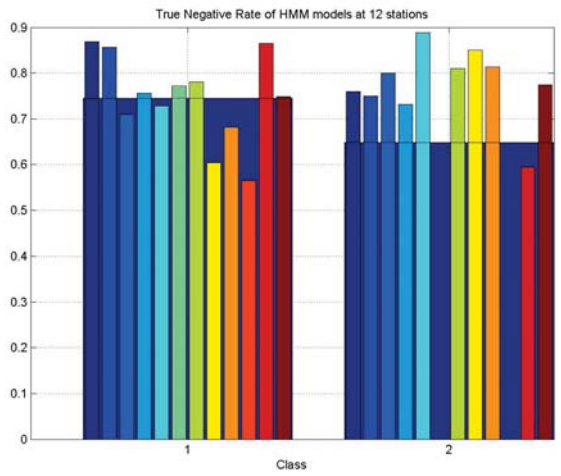


Fig. 15: TNR of the HMM model for each of the 12 station, in the 2-class framework.

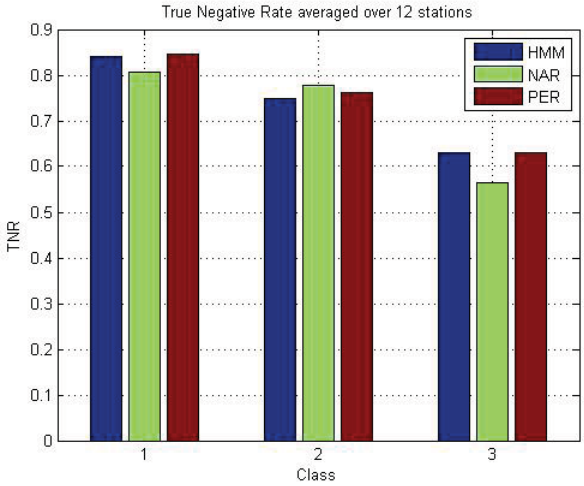


Fig. 17: TNR for the 3-class framework averaged over twelve stations.

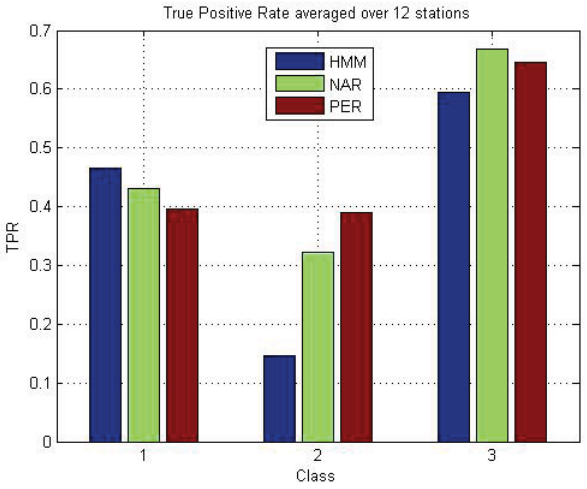


Fig. 16: TPR for the 3-class framework averaged over twelve stations.

7. Conclusions

In this paper a feature based strategy to cluster solar radiation daily patterns was presented, which allows associating to the original solar radiation time series a time series of classes. This allows to perform some useful statistics which would be otherwise not possible to make, such as the class weight and the persistence analysis. Furthermore, the paper has addressed the problem of one-day ahead prediction of the class. To this purpose two different approaches were considered, namely the HMM and the NAR approach. Results show that, at the present stage of this work, the prediction models are effective for a 2-class framework only. Work is still in progress to improve these results.

8. Acknowledgements

The research was supported by the Universitá di Catania under the grant FIR 2014.

References

[1] L. Fortuna, G. Nunnari, S. Nunnari, Nonlinear modeling of solar radiation and wind speed time series, SpringerBrief in Energy, ISBN 978-3-319-38763-5.
[2] T. Soubdhan, R. Emilion, R. Calif, Classification of daily solar radiation distributions using a mixture of dirichlet distributions, Solar Energy 83 (2009) 1056–1063. doi:10.1016/j.chaos.2008.07.020.
[3] M. Nijhuis, B.G. Rawn, M. Gibescu, Classification technique to quantify the significance of partly cloudy conditions for reserve requirements due to photovoltaic plants, Proceedings of the 2011 IEEE Trondheim PowerTech Conference, 2011.
[4] L. Fortuna, G. Nunnari, S. Nunnari, A new fine-grained classification strategy for solar daily radiation patterns,Pattern Recognition Letters, 2016, DOI: 10.1016/j.patrec.2016.03.019.
[5] S. Wilcox, National Solar Radiation Database 1991–2010 Update - Users Manual, Technical Report NREL/TP-5500-54824, 1–479, 2012. ftp://ftp.ncdc.noaa.gov/pub/data/nsrdb-solar/documentation-2010/.
[6] P. Ineichen and R. Perez, A New airmass independent formulation for the Linke turbidity coefficient,Physica A,2002,73,151–157.
[7] R. Perez, A New Operational Model for Satellite-Derived Irradiances-Description and Validation, Solar Energy, 2002,73, 207–317.
[8] J. Geweke, S. Porter-Hudak, J. Time Series Analysis 4 (1983) 221.
[9] R. Weron, Estimating long range dependence finite sample properties and confidence intervals, Physica A 312 (2002) 285–299.
[10] T. Kohonen, Self-Organizing Maps, 1995.
[11] T. W. Liao, Clustering of time series data - a survey, Pattern Recognition 38 (2005) 1857–874.
[12] L. A. Zadeh, Fuzzy sets, Information and Control 8 (1965) 338–353.
[13] T. Kohonen, Self-Organizing Maps, 1995.
[14] T. W. Liao, Clustering of time series data - a survey, Pattern Recognition 38 (2005) 1857–874.
[15] L. R. Rabiner, A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. Proceedings of the IEEE, 77 (2), 257Ű286, 1989.
[16] A. J. Viterbi, Error bounds for convolutional codes and an asymptotically optimum decoding algorithm, IEEE Transactions on Information Theory 13 (2), 260Ű269. doi:10.1109/TIT.1967.1054010.

Merging Event Logs for Process Mining with Hybrid Artificial Immune Algorithm

Yang Xu [1], **Qi Lin**[1], **Martin Q. Zhao**[2]
[1]School of Software Engineering, South China University of Technology, Guangzhou, China
[2]Computer Science Department, Mercer University, Macon, Georgia, USA

Abstract - *Current process mining techniques and tools are based on a single log file. In an actual business environment, however, a process may be supported by different computer systems. So it is necessary to merge the recorded data into one file, and this mission is still challenging. In this paper, we present an automatic technique for merging event logs with an artificial immune algorithm combed with simulated annealing. Two factors that we used in the affinity function, occurrence frequency and temporal relation can express the characteristics of matching cases more accurate than some factors used in other solutions. By introducing simulated annealing selection and an immunological memory to strengthen the diversity of population, the proposed algorithm has the ability to deal with local optima due to pre-mature problem of artificial immune algorithms. The proposed algorithm has been implemented in the ProM platform. The test results show a good performance of the algorithm in merging logs.*

Keywords: Process Mining, Event Log Merging, Artificial Immune

1 Introduction

Process mining makes use of recorded historical process data in the form of so called event logs to discover and analyze as-is process models [1]. In an intra- or inter-organizational setting, log data are distributed over different sources, each of which encompasses partial information about the overall business process. However, currently available process mining techniques, such as Heuristics Miner [2], Generic Miner [3], Fuzzy Miner [4], and Conformance Checking [5] require these data first to be merged into a single event log. Merging logs for an extended analysis of the entire process becomes a challenging task.

The most remarkable research on merging the log files is made by Claes [6], in which an artificial immune system is used. Their method assumes the existence of an identical case ID among various logs, or a case ID as an attribute in each event log with which events in different logs related to the same case can be matched. When two traces have the same case identifier, it is easy to establish the correlation directly. In real-life processes, however, each sub-process often employs different case ID. It is not easy to keep the case ID same in different logs. In addition, as with many applications of genetic algorithm, the artificial immune algorithm also has a problem of low performance due to tendency to converge towards local optima.

In this paper we present an automatic technique for merging event logs, which is based on a hybrid artificial immune algorithm combined with simulated annealing. In the method, we optimize the affinity function with two factors: occurrence frequency of execution sequences and temporal relations between process instances. An immunological memory is used to keep solutions with highest affinity score and apply simulated annealing to strengthen the diversity of later generations so as to alleviate the problem of local optima.

The rest of the paper is structured a follows. Section 2 clarifies the concepts about merging log data. Section 3 discusses related work. Our approach and the log merging algorithm are introduced in section 4. Section 5 evaluates our approach. Finally, conclusions are given in Section 6.

2 Related concepts

Before we introduce our approach, we first need to clarify some concepts related to event log merging.

Definition 1 (**Process**) A process is a tuple $P = (A, I, O, A_0, A_e)$, where,

- A is a finite set of activities,
- $I: A \rightarrow \mathcal{P}(\mathcal{P}(A))$ is the input condition function. $\mathcal{P}(A)$ denotes the powerset of some set A.
- $O: A \rightarrow \mathcal{P}(\mathcal{P}(A))$ is the output condition function,
- $A_0 \in A$ is start point of the process, $I(A_0) = \{\emptyset\}$,
- $A_e \in A$ is end point of the process, $O(A_e) = \{\emptyset\}$

For a process $P' = (A', I', O', A'_0, A'_e)$, if $A' \subseteq A$, and $I' \subseteq I$, $O' \subseteq O$, then call P' is a sub-process of P, denoted as $P' \subseteq P$.

Definition 2 (**Execution sequence**) Given a process $P = (A, I, O, A_0, A_e)$, a sequence $\sigma \in A^*$ is called a execution sequence, if and only if, for $n \in IN$, there exist $a_1, \cdots, a_n \in A$ and $a_1 = A_0$, $a_n = A_e$, such that $\sigma = a_1 \cdots a_n$, and, for

all i with $1 < i < n$, I_i and O_i are the input condition function and output condition function of a_i respectively, $I_i \subseteq \mathcal{P}(\mathcal{P}(\{a_1, \cdots, a_i\}))$, and $O_i \subseteq \mathcal{P}(\mathcal{P}(\{a_i, \cdots, a_n\}))$.

Definition 3 (**Log schema**) A log schema S is a finite set $\{D_1, \cdots, D_n\}$, where D_i with $(i = 1, \cdots, n)$ is called as attribute.

Definition 4 (**Event log**) Given a log schema S for a process P, an event log is a tuple $L = (S, P, E, \rho)$, where, $E \subseteq D_1 \times D_2 \times \cdots \times D_n$ is a finite set of events $\{D_1, \cdots, D_n\}$, and ρ is a mapping from the set E to the process P, that is $\rho: E \rightarrow A$.

In the process mining domain, an executed activity of a process, called an ***event***, is recorded in event log. An event $(d_1, \cdots, d_n) \in D_1 \times D_2 \times \cdots \times D_n$ is an interpretation over the set of activities A associating the log schema, i.e. an instance of the log schema. Actually, the event is the smallest unit in event log. A record in log is the description of an event with a group of attribute values, for example, execution time and executor of an activity, which are defined in a log schema. To be convenient, an event log can be briefly regarded as a set of events, denoted by $L(P)$. The denotation of $e.d$ represents the value of attribute d while e denotes an event.

Definition 5 (**Case**) Given an event log L, a set of events $\omega \subseteq E$ is called a case of L, denoted as $\omega \in L$, if and only if, the following conditions are met,

- Each event in ω appears only once. That is, for event $e_i, e_j \in \omega$ with $1 \leq i, j \leq |\omega|, e_i \neq e_j$,
- The events in ω are ordered, and
- ω is one of the instances, or the actual execution of an execution sequence.

A case also has attributes. For example, the order of events in a case is an attribute of the case. The duration of a case from the time of the start event to the end event is another case attribute. The denotation of $\omega.a$ represents the value of attribute a of the case ω.

Obviously, an event log can also be regarded as a finite set of cases. When we call a log as event log, it means that the log has been structured for process mining from the raw log, and all the cases have been identified.

Definition 6 (**Occurrence frequency**) Given a process P and its event log L, let T be a finite set of all execution sequences of P, $\mathcal{F}: T \rightarrow \text{IN}$ is a mapping about T over L. For $\forall \sigma \in T$, $\mathcal{F}(\sigma)$ is the number of occurrences of σ.

Definition 7 (**Mergable log**) Let two processes P_1, P_2 with $P_1 \subseteq P$, $P_1 \subseteq P$, and $L(P_1) \cap L(P_2) = \emptyset$. $L(P_1)$ can be merged with $L(P_2)$, if $\forall \omega^{(P_1)} \in L(P_1)$, $\exists \omega^{(P_2)} \in L(P_2)(\omega^{(P_1)} \cup \omega^{(P_2)} \in L(P))$.

Basically, event log merging consists of two steps: (i) correlate cases of both logs that belong to the same process execution and (ii) sort the activities in matched cases into one case to be stored in a new log file. The main challenge is to find the correlated cases in both logs that should be considered related to the same entire case. In this paper we limit ourselves to this challenge and we take the following assumptions:

- Logs from various systems can be preprocessed to a uniform format [7].
- All cases in logs have been identified [8], and
- Every event in logs has the attribute of reliable and comparable timestamp such that events can be easily sorted according to the values of timestamps. That is, $(\omega, \leq_{timestamp})$ is a partially ordered set, for $e_i, e_j \in \omega, 1 \leq i < j \leq |\omega|$, $e_i \leq_{timestamp} e_j$.

3 Related works

Merging logs is the problem of log pre-processing for process mining. Other researches on pre-processing of logs focus on uniform format [7] and event correlation [8], while the actual merging process has not been widely addressed.

Several approaches have been proposed in recent years. Rule-based log merging method [9] provides a tool with which a set of general merging rules are used to construct a set of specific matches between cases from two event logs. However, the tool only provides the descriptive language for the rules. The rules and decisions for specific logs have to be made by users who need to have enough knowledge about both the business process and the rule language. Text mining-based merging method [10] uses text mining techniques to merge logs with many to many relations. In this approach, similarity of attribute values is calculated and cases are matched according to the assumption that matching cases would have more common words in their attribute values than non-matching cases. The approach still has to face the situation in which identical word has possibly different semantics in two logs.

Artificial immune system-based method [6] is the most remarkable approach for merging logs. This approach exploits the artificial immune algorithm [11] which is inspired by the biological immune system to merge two mergeable logs. Judging whether two cases from two different logs belong to the same process execution becomes a process in which the strength of the binding of antigen and antibody is evaluated. The strength is related to the affinity between an antigen and the antibody in the binding, i.e. between two cases. When the affinity reaches certain threshold value, the two cases can be regarded as belonging to the same process execution. The underlying principle is the Clonal Selection Principle, which requires the matching process undergo iterations of clonal selection, hypermutation and receptor editing until a certain stop condition is met. The algorithm starts from a random population of solutions. Each solution is a set of matches

between the cases in both logs. The affinity of each matching is calculated and the total is summed for the solution. The higher the affinity value of a matching, the higher the likelihood for that matching being cloned, the fewer the chances for being mutated and edited. When a stop condition is met through the evolution of several generations, the solution with highest affinity is the proposed solution.

Obviously, the calculation of affinity is a very important impact factor on the performance of the algorithm. The authors assume identical case ID as an attribute in the event log with which the affinity is calculated. Actually, it is meaningless for the affinity calculation when different case IDs are exploited in the real life logs. On the other hand, the artificial immune algorithm also has a problem of tending to converge towards local optima. No discussion about the problem is present in their papers.

In summary, the problem of merging logs for process mining has not been addressed so far. Our work in this paper is trying to fill the gap in this research area.

4 Merging event log

4.1 Overview

Basically, correlating cases from two logs is the process in which finding the best solution among all possible solutions with matched cases. We believe that using only one factor, e.g. timestamp of events or case id, to judge whether two cases from the different logs belonging to the same process execution is not reliable due to the complexity of business processes and their execution context, as well as the imperfect logs caused by operational faults of information systems and other unexpected events. For this kind of search and optimization problems, evolutionary algorithms, such as genetic algorithm and artificial immune system, are the appropriate selection. In this paper, we choose artificial immune system with an immunological memory as the foundation. In this section we describe the key steps of our log merging approach.

The approach starts from a random population of solutions. The solutions are sorted according to their affinities. The top p percent solutions with higher affinity are selected to construct an initial population for subsequent immune process. Every generation of population has to go through four steps: clonal selection, hypermutation, annealing operation and diversity strengthening. The algorithm then iterates over these steps until a stop condition is met. Here, we set a fixed amount of iterations as the stop condition.

The solutions in each population are sorted according to their affinity. The top ranking solutions with the highest scores are selected (cloned) to construct the next generation. Then, each solution is mutated to build a new population. The amount of mutations on each solution also depends on its affinity: the higher the affinity, the less mutation.

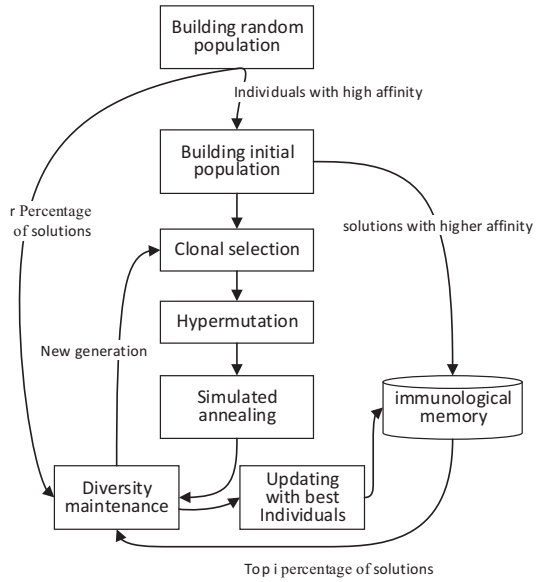


Fig. 1. The processing framework of merging logs

Like genetic algorithm, premature convergence may often occur in artificial immune search and make the merging process slow down. Diversity maintenance in every generation is the key for avoiding premature convergence or persistent degradation. We introduce simulated annealing in artificial immune algorithm to guarantee that not only better solutions with higher affinity than last generation are kept, but also some deteriorative solutions at a certain probability are accepted to generate the new population. On the other hand, we set an immunological memory to keep solution with the highest affinity in every generation. When building a new generation, the solution in the memory is selected to strengthen the diversity of antibodies.

The most important design choice in the algorithm is the affinity function, which determines the affinity of a solution. The higher the affinity, the greater the probability of merging two logs. Every step is influenced by the affinity as the evaluation of affinity is used throughout the whole algorithm.

In the affinity function, two important factors related to attributes of case, *occurrence frequency* and *temporal relation*, are used. If two cases from different logs belong to the same whole process execution, the occurrence frequencies of their corresponding *execution sequences* are statistically equivalent. If the difference between *occurrence frequencies* of two execution sequence exceeds a threshold value, the corresponding cases are not considered from the same process execution. The factor of *temporal relations* describes the time overlap between two cases. In this paper, we assume that (1) before merging two logs, we know which corresponding process starts first; and, (2) if two processes whose logs can be merged, one process that starts before the beginning of the other process triggers the latter. It should have some time overlap between the processes. Hence, the time overlap can be used to check whether two cases are matched on time property, described in detail below.

4.2 Affinity function

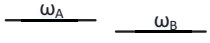
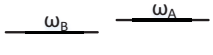
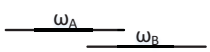
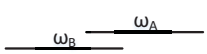
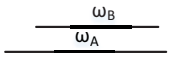
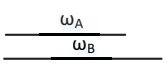
Because more than one factor can indicate that two cases should be matched, the affinity is evaluated with a number of indicator factors.

$$f = \alpha_1 \sum AOF_i + \alpha_2 \sum OLT_j + \alpha_3 \sum EAV_k \quad (1)$$

Here, AOF_i indicates (approximate) equivalent occurrence frequency between two execution sequence of two logs. Given case $\omega_A \in L(P_1)$, $\omega_B \in L(P_2)$ and $P_1, P_2 \subseteq P$, if ω_A and ω_B are matched, i.e. $\omega_A \cup \omega_B \in L(P)$, then the Occurrence frequency of the execution sequence of ω_A is equivalent to, or close to that of ω_B. This factor has a greater positive effect on the affinity value, indicating that two logs have greater probability to be mergeable when the situation appears more.

OLT_j indicates the temporal relations between two processes. If two logs are mergable, one corresponding process should be triggered by the other. The triggered one should start after the beginning of the other. Therefore, there is some time overlap between the two cases. Table 1 shows the types of temporal relations between two cases. Note that we have to make a decision on which case starts earlier before the temporal relation is used to calculate the affinity. In table 1, ω_B is triggered by ω_A.

Table 1. Temporal relations between two cases

Relations	Illustrations	Match	Comments
$\omega_A < \omega_B$		N	No overlap - No relation of triggering between ω_A and ω_B
$\omega_B < \omega_A$		N	
$\omega_A \preccurlyeq \omega_B$		Y	Partial overlap - ω_B is triggered by ω_A.
$\omega_B \preccurlyeq \omega_A$		N	Partial overlap - ω_A is triggered by ω_B.
$\omega_A \ni \omega_B$		Y	Full contain - ω_B is triggered by ω_A and ends before ω_A.
$\omega_B \ni \omega_A$		N	Full contain - ω_A ends before ω_B.
$\omega_A \parallel \omega_B$		N	Parallel between ω_A and ω_B

In our affinity calculation, only two temporal relations, Partial overlap and Full containment, make contributions. It seems that we only need to take into account the situations of $\omega_A \preccurlyeq \omega_B$ and $\omega_A \ni \omega_B$, since only these two indicate the matching between ω_A and ω_B. However, it is possible that the relations $\omega_A \preccurlyeq \omega_B$ and $\omega_B' \preccurlyeq \omega_A'$ in which ω_A and ω_A' have the same execution sequence as well as ω_B and ω_B', are both appear in the relation statistics. In this situation, the rule that the relations $\omega_A \preccurlyeq \omega_B$ and $\omega_A \ni \omega_B$ increase the affinity while $\omega_B \preccurlyeq \omega_A$ and $\omega_B \ni \omega_A$ decrease the affinity.

EAV_k indicates matching values of event attributes between two logs. In real life processes, it often happens that some values (e.g. "invoice code") are passed from event to event between two cases belonged to two different logs but identical whole process. Hence, we can calculate the amount of matches of event attribute values to indicate the affinity of two logs.

The α_1, α_2 and α_3 are the weight of the factors with $\alpha_1 + \alpha_2 + \alpha_3$=1. Every weight can be adjusted by the user with his knowledge of business process to adapt the algorithm to concrete problem.

Of course, some other factors of case attributes or event attributes, such as identical case ID, can be added into the affinity function. Some factors representing business properties are allowed to be considered to adapt the evaluation of the affinity to actual business. These factors need to be designed by business specialists.

4.3 Immunological memory

The immunological memory is set to keep solutions with the highest affinity score in every population. The solutions in the memory are selected to generate next generation of population. The initialization of the memory occurs when initial population is constructed. The top t percent solutions are selected into the memory. And these solutions are updated with those with much higher affinity when new candidate solutions produced by mutation and picked up by the simulated annealing.

4.4 Simulated Annealing

Simulated annealing is a probabilistic method based on the physical process of metallurgical annealing for approximate global optimization in a large search space [11]. Here, we exploit it to make the artificial immune merging jumping out from the trap of local optima. When a population undergoes mutation, we evaluate the new candidate solutions and get the best one whose affinity is the highest in the population. If it has a higher affinity value than the highest from the previous population, the simulated annealing accepts it. Otherwise, it is accepted into this new population with a certain probability calculated according to the equation (2).

$$Pr = exp(|\Delta Z|/kT) \quad (2)$$

Here, ΔZ is the affinity difference between the new candidate solution and the best solution in last generation. T represents the current temperature, and k is a constant. The algorithm for accepting candidates is shown in algorithm 1.

Algorithm 1: Accept new populations by simulated annealing

```
Input: G(i) //the current population
       P(i) //the best solution of the current population G(i)
       B(i) //the worst solution in the immunological memory Bank[]
outout: G(i+1) // next population
T ← affinity(P(i)) - affinity(B(i)); // initial temperature
T_min ← 0.0;  //temperature for stopping
while G(i) do
   calculate each individual's affinity in G (i);   // equation (1)
   if P(i) > B(i) do
      update Bank[] with the best individual in G(i);
   if terminal condition is true then
      return
   G '(i+1);    //clone and mutation for new population
   P'(i+1);    //the best individual in G '(i+1)
   if ( T > T_min ) do
      ΔZ  ← affinity(P'(i+1)) - affinity (P(i));
      if ( ΔZ>= 0 ) do
         G(i+1) ←G '(i+1);   //accept new population
      else
         if ( exp(ΔZ / kT) > random(0,1) ) do
            G(i+1) ←G '(i+1);    //accept new population
            T ← r * T;     // temperature decrease
         else
            G(i+1) ← G(i);    // reject new population
```

4.5 Diversity maintenance

The solutions are selected by simulated annealing into the next population. This new population has to contain as many elements as the previous one and for this reason new solutions are picked from two other sources. One is the solutions from the random population. The other is the solutions in the immunological memory. All solutions of the random population and the immunological memory have a chance to be selected for the new generation, but the solutions with a higher fitness score still get a higher chance than the solutions with a lower fitness score. The amount of solutions from the three sources is calculated with equation (3).

$$size_{pop} = N \times n + I \times i + R \times r \qquad (3)$$

In equation (3), N represents the amount of solutions in new population accepted by the simulated annealing. I represents the amount of solutions from the immunological memory. And R represents the amount of solutions from the random population. The n, i, r are the are the percentages of N, I, R respectively, and can be decided by users.

4.6 Complexity analysis

The time overhead of our approach depends on size of logs. We assume that there are n_1 and n_2 cases respectively in two logs, with an average of m events each case and an average of r attributes each event.

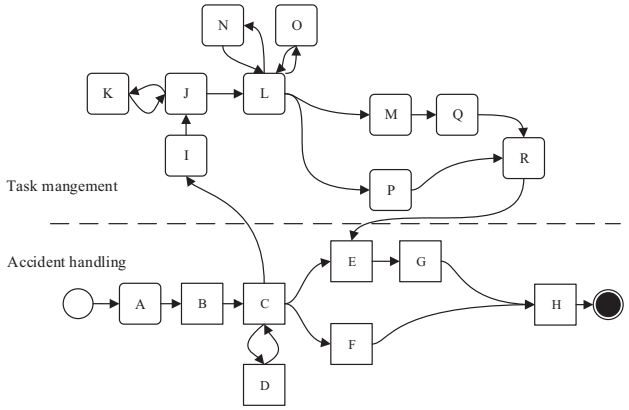
In the step of initial population, n solutions have to be built ($n = \max\{n_1, n_2\}$). The affinity of every candidate solution is calculated according to formula (1). The overhead of the initialization is $T_1(n) = O(nmr)$, i.e. $O(n)$..

Assume the population evolves s generations before stop. The overhead in clone and mutation steps including the affinity evaluation is $T_2(n) = O(sk^2n^2)$. Here, k is the number of mutation operation. If each solution in every generation only mutate once which is the best case, the overhead is $T_2(n) = O(sn^2)$ with $k = 1$. In the step of generating new population, the overhead of simulated annealing selection, affinity evaluation and update of immunological memory are all $O(n)$.

In summary, the complexity of the algorithm is $O(sk^2n^2)$ which depends on the amount of cases n, the amount of mutation operation k and the number of generation s. Ideally, the complexity is $O(sn^2)$.

5 Evaluation

The proposed algorithm has been implemented and evaluated in a controlled experiment using logs from two information systems: incident management system and task management system. These two systems support the implementations of processes of accident handling procedure (for short P_A) and task management procedure (for short P_B), which are sub-processes of Incident Lifecycle Management for IT operation and maintenance. The model we used in our experiments is shown in figure 2.



A: Register Acc. B: Dispatch Acc. C: Treat Acc. D: suspend Acc. E: Solve Acc. F: Cancel Acc. G: Confirm Acc. H: Close Acc. I: Create task J: Assign task K: Suspend task L: Execute task M: Complete task N: Transmit task O: Request assist. P: Cancel task Q: Evaluate task R: Close task

Fig. 2. The processing framework of merging logs

In the model, after an accident has been designated to somebody to handle (activity node C in P_A), P_B is triggered with a new task being created (activity node I in P_B). Note that when a task is closed (activity node R in P_B) , P_B provide a feedback to the activity E in P_A to continue the accident processing. Therefore, the temporal relationship between cases of the two processes is Full containment.

Table 2. Information of logs

Group	Logs	#Cases	#Event
G1	log1 vs. log2	1024 vs. 1024	5790 vs. 7942
G2	Log3 vs. log4	2048 vs. 2048	11598 vs. 15847
G3	Log5 vs. log6	4096 vs. 4096	23401 vs. 31903

In our experiments, we design three groups of tests, each group having two logs for merging: log1 vs. log2, log3 vs. log4 and log5 vs. log6. The logs in three groups are different in log characteristics, such as the number of cases and the number of events, as shown in table 2. We compare the Claes's algorithm (AIA) and the proposed algorithm (SA+AIA) in two aspects: merging quality and merging efficiency. We use the success rate of merging to measure merging quality. The success rate of merging is a ratio between number of successful match of cases C_s and all of cases in two logs C_a and C_b , as shown in equation 4.

$$R_s = 2\sum C_s / \left(\sum C_a + \sum C_b\right) \qquad (4)$$

The execution time is a significant indicator of the merging efficiency. Due to the stochastic nature of the algorithms, the experimental results are assessed with average number of successful matched cases (ACMC) and average execution time (AET). The ACMC and AET of each test are averaged over 3 independent runs, in which we use the same logs and the same setting. The tests run on a 6GB RAM 2.2GHz laptop. The parameters of the algorithms are set as follow: the size of random population is 100; the percentage for initial population: 60%; the weight of the factors α_1, α_2 and α_3 are 40, 30, 30; and the constant k for annealing is 0.95.

Table 2. Test results

	ALG	ACMC	Merging success rate	#Generations before stop	AET (secs)
G1	AIA	949	92.68%	10000	92
	SA+AIA	957	93.36%	6367	72
G2	AIA	1883	92.03%	10000	197
	SA+AIA	1907	93.12%	7362	153
G3	AIA	3735	91.12%	10000	409
	SA+AIA	3785	92.41%	7299	328

In general, the results indicate that the proposed algorithm performs better than AIA algorithm with merging success rate of merging of at least 90%, as shown in figure 3. The results in Table 3 show that the proposed algorithm has reduced the number of iterations significantly over the AIA algorithm. The time overhead is reduced by an average of 21% over the AIA algorithm, as shown in figure 4. This means that the proposed algorithm can accelerates convergence to the global optimum. Note that, however, the performance of the algorithm is sensitive to the amount of logs. The success rate has a downward trend and the time overhead increases with the increasing of the amount of logs. It is reasonable to believe the trend will be more obvious with the increasing complexity of process and size of logs. Figure 5 shows the mining result based on the merged log with Alpha method in ProM. The mined model is the same as the model shown in Figure 2.

In summary, the proposed approach has better performance than the exsiting AIA method in merging logs.

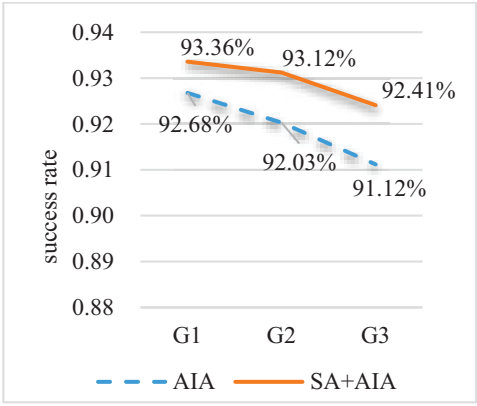


Fig. 3. Test Results of Merging quality

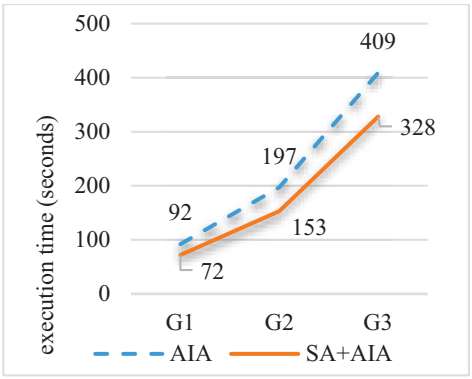


Fig. 4. Test Results of Merging efficiency

6 Conclusions

In this paper we presented a solution to merging log files from different systems into a single file so that the existing process mining techniques can then be used. The proposed algorithm is inspired by artificial immune algorithms and simulated annealing algorithms. An affinity function is the key in every step in this algorithm. The set of factors used in the affinity function, occurrence frequency and temporal relation can express the characteristics of matching cases more accurate than some factors, e.g. case ID used in other solutions. Another unique feature of the proposed algorithm is its ability to deal with local optima due to pre-mature problem of artificial immune algorithms by introducing simulated annealing selection and immunological memory to strength the diversity of population. The proposed algorithm has been implemented as a plugin into the ProM platform. The test result shows a good performance of the algorithm in merging logs.

It has been found that the performance of the proposed approach, especially time overhead, is sensitive to the size of the log files, same as other approaches. In real life business, massive logs are produced every day. It is a big challenging to improve the merging efficiency for massive log data. Distributed or parallel merging approach for massive log will be explored in our future research.

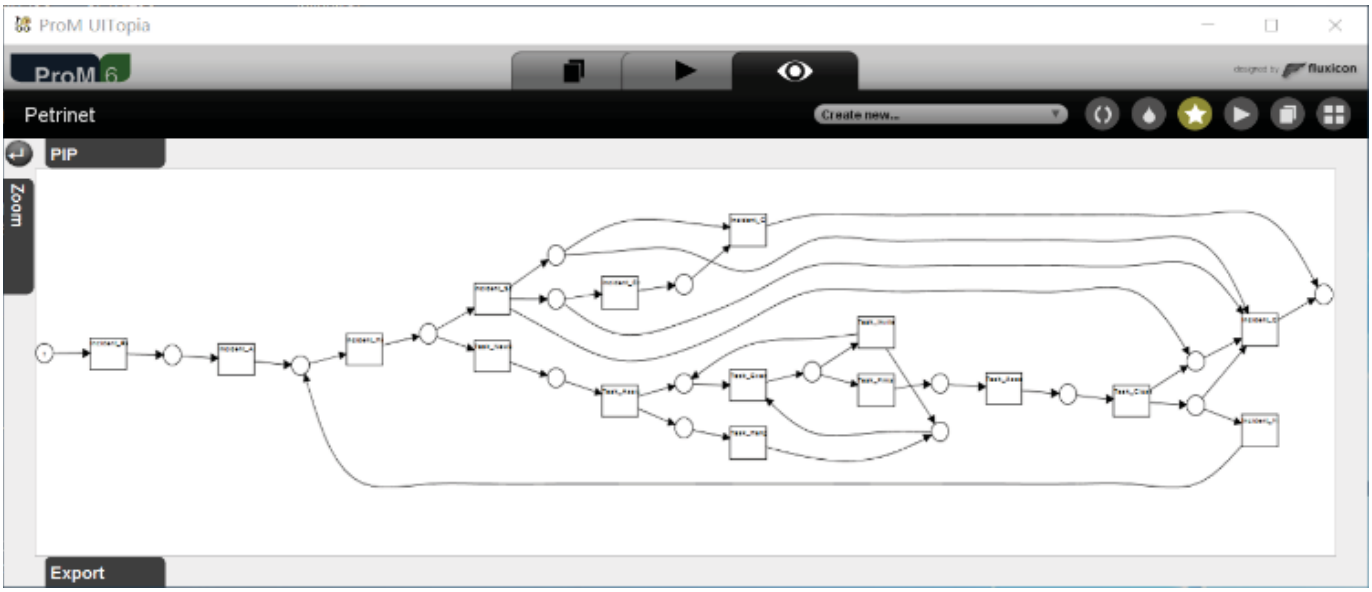


Fig. 5. The Model mined in ProM

Acknoweledge

This work was supported in part by the National Natural Science Foundation of China under Grant 71090403, the Guangdong Provincial S&T Planned Projects under Grant 2014B090901001/2015B010103002 and Special Funds on "985 Project" Disciplinary Construction in School of Software Engineering of South China University of Technology under Grant x2rjD615015III.

References

[1] W. M. P. van der Aalst, "Process Mining: Discovery, Conformance and Enhancement of Business Processes". Springer Berlin Heidelberg, 2011.

[2] A. J. M. M. Weijters , J. T. S. Ribeiro. "Flexible Heuristics Miner". Proccedings. of 2011 IEEE Symposium on Computational Intelligence and Data Mining, Paris, pp. 310—317, April 2011.

[3] A. K. A. de Medeiros, A. J. M. M. Weijters, W. M. P. van der Aalst. "Genetic process mining: an experimental evaluation". Data Mining and Knowledge Discovery, vol. 14, issue 2, pp. 245—304, April 2007.

[4] B. F. van Dongen, A. Adriansyah. "Process Mining: Fuzzy Clustering and Performance Visualization". Business Process Management Workshops, Springer Berlin Heidelberg, pp.158—169, September 2010.

[5] W. M. P. van der Aalst, A. Adriansyah, B. F. van Dongen, "Replaying history on process models for conformance checking and performance analysis", Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, vol. 2, issue 2, pp.182—192 , April 2012

[6] J. Claes, G. Poels, "Merging Computer Log Files for Process Mining: an Artificial Immune System Technique", Procceding of the 9th International Conference on Business Process Management Workshops, France, pp. 99—110, August 2011.

[7] H. M. W. Verbeek, Joos C. A. M. Buijs, Boudewijn F. van Dongen, W. M. P. van der Aalst. "XES, XESame, and ProM 6". Information Systems Evolution, Springer Berlin Heidelberg, pp. 60—75, June 2010

[8] H.R. Motahari-Nezhad, R. Saint-Paul, F. Casati, B. Benatallah, "Event correlation for process discovery from web service interaction logs ". The International Journal on Very Large Data Bases, Springer New York, vol.20, no.3, pp. 417—444, June 2011

[9] J. Claes, G. Poels, "Merging Event Logs for Process Mining: A Rule Based Merging Method and Rule Suggestion Algorithm". Expert Systems with Applications, vol.41, issue.16, pp.7291—7306, November 2014.

[10] L. Raichelson, P. Soffer. "Merging Event Logs with Many to Many Relationships". Business Process Management Workshops, the series Lecture Notes in Business Information Processing, Springer International Publishing, vol. 202, pp.330—341, April 2015.

[11] E. K. Burke, G. Kendall, "Search Methodologies-Introductory Tutorials in Optimization and Decision Support Techniques (2nd Ed) ". Springer New York, 2014.

Forecasting Movements in Oil Spot Prices using Data Mining Methods

M.E. Malliaris [1], **A.G. Malliaris**[2]
[1]Information Systems & Supply Chain Management Dept., Loyola University Chicago, Chicago, IL, USA
[2]Economics and Finance Departments, Loyola University Chicago, Chicago, IL, USA

Abstract -*This paper uses information about several variables related to oil fundamentals to predict the direction that the spot oil price will move in the next week. The data, was downloaded from the Energy Information Administration and spans the time from 2001 through early 2016. It is divided into four periods. We look at both the variable sensitivity and the model ability to forecast. We find that the variables' relationships alter over the periods. By using two artificial intelligence methodologies, decision trees and support vector machines, and only forecasting when they agree, we have a much better chance of being correct in identifying next week's directional move of oil price.*

Keywords: *Decision trees, Support vector machines, oil direction, forecasting*

1 Introduction

The price of oil plays an important role in the U.S. economy for many reasons. First, the price of oil and its volatility influence both the producer and the consumer price indexes and other measures of inflation. Current inflation also influences expected future inflation and interest rates. Second, the oil industry is a dynamic sector of the U.S. economy for the employment it generates, the technology it develops and its impact on other sectors such as transportation, industrial products and research and development. Finally, oil related products generate substantial tax revenues, part of which finance the U.S. highways system. Thus, understanding what moves oil prices and being able to forecast future trends has received great attention. Representative papers that document oil forecasting research can be seen in [1, 2, 3, 4, 5, 6, 7]. In this paper, we are similarly interested in forecasting the price of oil, but our interest is not in terms of time series methodology but rather the use and evaluation of data mining techniques. We also use structural breaks, or periods, to retrain the models.

2 Data and Periods

The entire data set spans the period from mid-November 2001 through the end of February 2016 and all values are weekly. Values for the five variables were downloaded from the Energy Information Administration (EIA) at http://www.eia.gov/dnav/pet/pet_sum_sndw_dcus_nus_w.htm. These variables are all based on oil and include a spot price for oil and proxies for the supply and demand for U.S. oil. The spot price is for West Texas Intermediate crude oil from Cushing Oklahoma. Demand is rather stable and changes slowly over time and we use the stock of oil as a proxy. When demand changes for seasonal reasons (summer traveling) stocks are drawn down to meet the increased demand. The stock of oil, excluding the strategic petroleum reserves, includes the inventories stored for future use and is reported in thousands of barrels on the last day of the week. Net imports are in thousands of barrels per day and include oil from the 50 states, the District of Columbia and U.S. possessions and territories. Crude oil supply is in thousands of barrels. Its components include field production, refinery production, imports, and net receipts calculated on a PAD district basis. Production is in thousands of barrels per day. The quantities are estimated by state and summed to the PADD and then the U.S. level. The paths of these variables are shown in Figure 1.

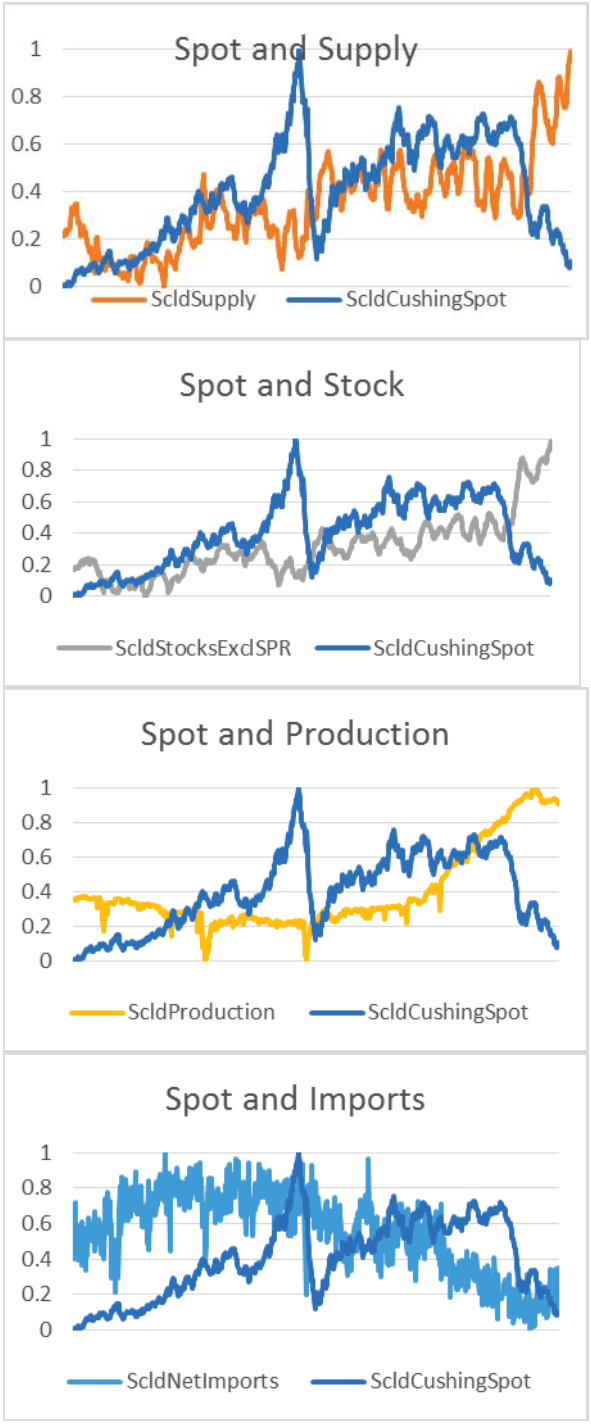
Table 1. Input and Target Variables

Derived Variables	Role	Example Value
ScldNetImports	Input	0.595
ScldSupply	Input	0.386
ScldStocksExclSPR	Input	0.324
ScldProduction	Input	0.234
ScldCushingSpot	Input	0.411
PerChgNetImp	Input	-0.015
PerChgSupply	Input	-0.013
PerChgStocksExclSPR	Input	-0.008
PerChgProduction	Input	0.002
PerChgCushingSpot	Input	0.011
DirNetImports	Input	Down
DirSupply	Input	Down
DirStocksExclSPR	Input	Down
DirProduction	Input	Up
DirCushingSpot	Input	Up
CushDirTp1	Target	Down

ISBN: 1-60132-431-6, CSREA Press ©

Using these five base variables, derived variables and a target were constructed. The derived variables included, for each of the base variables, a value scaled to be between zero and one, the percent change of the variable from week to week, and the direction the variable moved from week to week. The target variable for both model types is the direction that oil will move next week. All variables used, and their names, are shown in Table 1.

Figure 1. Cushing Oil Spot Price vs each other Variable, Scaled



This data set was divided into four periods, based on places where structural breaks have occurred, and models were built for each. Structural breaks in forecasting oil have been found important for accuracy [8, 9]. The periods are named Before, Crisis, After, and Recent. The dates used for each of the four periods are shown in Table 2. In the initial period, oil shows an overall steady positive increase in price. The Crisis period saw a steep increase in the spot price followed by an even steeper decrease. After the crisis, oil prices showed volatility with an overall path of a slight increase. In the Recent period, prices show a decreasing trend.

Table 2. Data Set time periods

Name	Beginning	Ending	#Weeks
Before	11/16/2001	8/31/2007	303
Crisis	9/7/2007	6/26/2009	95
After	7/3/2009	12/26/2014	287
Recent	1/2/2015	2/26/2016	61

Table 3. Correlations between variables per period, negative cells shaded

Before	*ScldNtImp*	*ScldSup*	*ScldStks*	*ScldPrd*
ScldNetImports	1.000			
ScldSupply	0.075	1.000		
ScldStocksExclSPR	0.309	0.887	1.000	
ScldProduction	-0.371	-0.449	-0.413	1.000
ScldCushingSpot	0.528	0.557	0.671	-0.787
Crisis				
ScldNetImports	1.000			
ScldSupply	-0.323	1.000		
ScldStocksExclSPR	-0.258	0.925	1.000	
ScldProduction	0.136	0.318	0.569	1.000
ScldCushingSpot	0.207	-0.698	-0.687	-0.261
After				
ScldNetImports	1.000			
ScldSupply	-0.089	1.000		
ScldStocksExclSPR	-0.419	0.600	1.000	
ScldProduction	-0.811	0.017	0.633	1.000
ScldCushingSpot	-0.309	0.108	0.270	0.346
Recent				
ScldNetImports	1.000			
ScldSupply	0.218	1.000		
ScldStocksExclSPR	0.302	0.894	1.000	
ScldProduction	-0.299	-0.137	-0.018	1.000
ScldCushingSpot	-0.616	-0.327	-0.349	0.655

Each of the periods contains a shift in the relationships among the variables. Table 3 indicates the positive and negative correlations in each period, with the negative correlations shaded. Notice, for example, that the correlation between the Cushing Spot price and Supply is positive in the Before period, negative in the Crisis period, positive in the After period, and negative in the Recent period. In order for a single model to remain stable and robust over time, the relationships among the variables also need to be stable. Because each of these variables has an impact on the price of oil, but this impact shifts throughout the periods, we build separate models on each period. After building these models, we look at the impact of the most important variables in each of the periods, and compare the forecasting results for the oil spot price movement.

3 Methods

Two data mining methodologies are used for this paper, decision trees and support vector machines. All models were built using IBM's SPSS Modeler 17.0 data mining software. A decision tree works in a stepwise fashion to successively divide the data into parts that are more single-valued on the target variable. In the beginning, all data is held in the root node. All models were built using IBM's SPSS Modeler 17.0 data mining software.

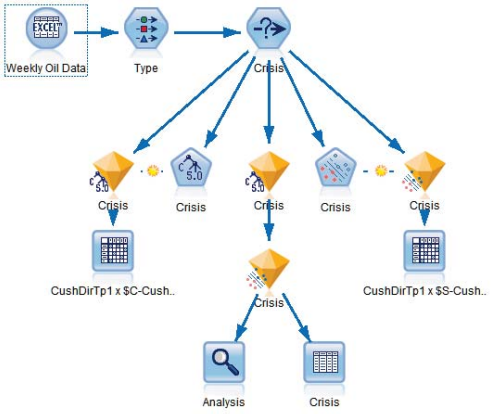
For the decision tree, the C5.0 algorithm was applied. This algorithm first splits the data set on the input field that gives the greatest information gain. That is, that provides the split with resulting groups that have higher percentages of a single value of the target than the original node did. Each of the nodes resulting from the first split are then split again, typically using a different field. This process continues until the nodes cannot have any possible split that could improve the model. Each terminal node forms a path from the root that describes a subset of the training data and generates a single value of the target for any new data that conforms to the path.

The second model used is the support vector machine methodology. This methodology applies a transformation to the input data that enables the two values of the target variable (in this case, Up and Down) to be separated by a plane so that each side of the plane has a single target value. Each row in the data set is graphed in n-dimensional space. Initially, the target variable values are not divisible. By applying a transformation that moves the data to an n+1 dimensional space, the target values form an arrangement that is separable. Modeler offers a choice of four transformations to the input data: radial basis function, polynomial, sigmoid, and linear. In this case, the polynomial transformation yielded the best results.

After each model has finished training, a sensitivity analysis is performed on all the variables. This generates a relative set of values for predictor importance. Various values of one variable are fed through the trained model while all other variable values are held fixed. The impact on the target variable is noted. This is repeated for each variable. Then, all variables are ranked according to the relative change to the target using the trained model. These predictor importance values range all sum to 1. Comparing these across models allows us to see the difference in how each of the models values the input of each of the variables, and how the variables impact the final target value.

Figure 2 illustrates the IBM Modeler data mining stream for the Crisis data period. The data set is read in with an Excel node and flows to a Type node where each field is assigned its role of input or target. The data for a particular period is then selected in the hexagon-shaped Select node, and sent to the pentagon-shaped modeling nodes, C5.0 and SVM, for training. Within the model training nodes, settings the affect the training run can be specified. This training process generates two diamond-shaped trained models which are shown as nuggets connected to their original training model. The results from the trained models are then sent to matrix nodes for evaluation of the output. In addition, copies of the trained models can be connected in succession, as shown here in the center of the figure. The data is sent through each of the trained models to an analysis node where the comparative forecasts can be studied. The data is also sent to a table node so that the forecasts can be exported to Excel for further analysis if desired.

Figure 2. Structure of the Modeler stream for one data set.



3.1 Decision tree results

The decision trees were developed separately for each of the four data sets. Figure 3 shows the trained decision tree maps for each of these data set periods. The Before and After maps are wider and deeper, but these data sets are also much larger than the Crisis and Recent sets are.

More important than just the complexity of the tree is how well they did in forecasting and explaining the paths to the forecasts. The results for forecasting accuracy are detailed in Table 4.

Figure 3. Decision Tree maps for each of the data sets

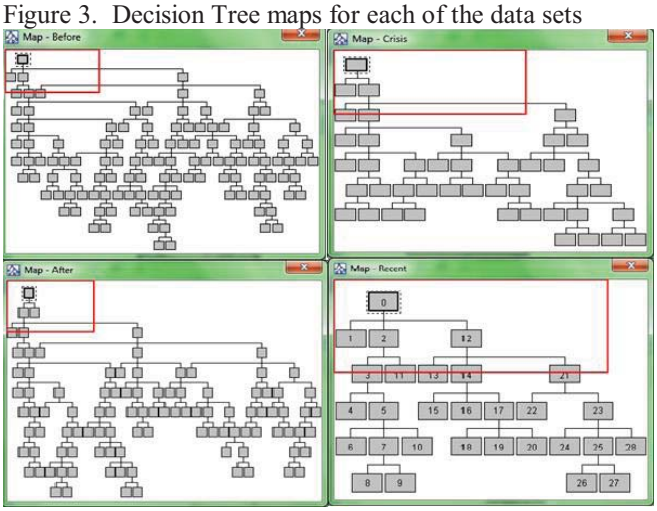


Table 4. Decision tree forecasting accuracy, correct percentages in bold.

		Actual Dir		Actual Dir		
Period		Down	% Cor	Up	% Cor	Total
Before	Dn	124	**89.2**	15		139
	Up	6		158	**96.3**	164
Crisis	Dn	40	**95.2**	2		42
	Up	6		47	**88.7**	53
After	Dn	135	**93.8**	9		144
	Up	8		135	**94.4**	143
Recent	Dn	33	**89.2**	4		37
	Up	2		22	**91.7**	24

In this table, we see that the forecasting accuracy for predicting that oil would move Down the next week varied from a low of 89.2% to a high of 95.2%. For the times that Up was predicted the worst period was correct 88.7% of the time, and the best was correct 96.3% of the time. The models are doing well in all periods and well in each direction of the forecast. We emphasize that these are "in sample" performances that do not guarantee similar results for "out of sample" performances. However, this forecasting accuracy demonstrates that the two techniques used were successful in identifying across 4 regimes the appropriate variables determining the in sample performance.

Table 5 shows the relative importance of each of the variables to the model trained in that period. These relative importance values sum to 1 within each model and are not related to model accuracy, but just to the way the trained model used the variables to forecast the target variable, the direction that oil would move in the following week. Blank cells indicate that the variable was not used. The top five variables for each model are shown in bold, and the most important variable is shaded. Since we saw in Table 3 that the correlations between the variables vary over time, we would expect variables to go in and out of importance in Table 5.

Table 5. Relative variable importance.

Variable	DT Before	DT Crisis	DT After	DT Recent
DirCushingSpot	**0.1242**	**0.1062**		0.0196
DirNetImports	**0.1126**	0.0541	**0.1247**	**0.3594**
DirProduction	0.0341	**0.2725**	**0.1234**	**0.1276**
DirStocksExclSPR	**0.0898**	0.0479	0.0580	0.0222
DirSupply	0.0881	0.0585	0.0214	0.0637
PerChgCushingSpot			0.0037	**0.1444**
PerChgNetImp	0.0635	**0.0775**	0.0000	**0.1149**
PerChgProduction		**0.1066**	**0.1429**	0.0099
PerChgStocksExclSPR	**0.0937**			
PerChgSupply	0.0484	**0.1905**	0.1117	
ScldCushingSpot	0.0498		**0.1441**	
ScldNetImports	0.0504	0.0657	0.0425	**0.1290**
ScldProduction	0.0396		**0.1356**	0.0093
ScldStocksExclSPR	0.0875	0.0205	0.0836	
ScldSupply	**0.1183**		0.0084	

The most important variable for each period is different, oil's direction is top in the Before period, the change in supply matters most during the Crisis period, After the crisis the change in production tops the importance list, and in the Recent period, the direction of net imports has over a third of the relative importance. We see similarities in the Before and After sets in that they use almost all of the variables. In the Crisis and Recent sets, a third of the variables are not used at all by the model.

Another way to think about the impact of an input is by summing the individual importance values of each variable related to the same base variable. For example, the three variables derived from the price of Cushing oil are oil's direction, oil's percent change in price, and its value as scaled from 0 to 1. These are represented by DirCushingSpot, PerChgCushingSpot, and ScldCushingSpot. Combining the variables in this way reduces the number of variables to four and allows us to see their overall impact regardless of the form in which they were presented to the models. Table 6 gives these summed values for the inputs related to each base variable.

When looked at as summed values we see that stocks of oil (excluding the strategic petroleum reserves) had the greatest overall influence in the Before period. Oil production was most important in both the Crisis and After periods, and net

imports climbed to the top spot with an overall relative importance of .6 in the Recent period.

Table 6. Sum of Dir, PerChg and Scld for each of the base variables, per period.

Variable	DT Before	DT Crisis	DT After	DT Recent
CushingSpot	0.1740	0.1062	0.1478	0.1640
NetImports	0.2265	0.1973	0.1672	**0.6033**
Production	0.0737	**0.3791**	**0.4019**	0.1468
StocksExclSPR	**0.2710**	0.0684	0.1416	0.0222
Supply	0.2548	0.2490	0.1415	0.0637

3.2 Support vector machine results

The support vector machines used a polynomial kernel, with the regularization parameter C set to 10 and epsilon at 0.1. The model accuracy on each period is shown in Table 6. Accuracy on weeks where the direction of oil was Down varied from a low of 90.8% to a high of 100%. For weeks where oil moved Up, accuracy varied from 92.6% to 100%. Lower accuracies occurred in the larger data sets.

Table 6. Support vector machine forecasting accuracy, correct percentages in bold.

		Actual Dir		Actual Dir		
		Down	%Cor	Up	% Cor	Total
Before	Dn	118	**90.8**	12		130
	Up	12		161	**93.1**	173
Crisis	Dn	45	**100**	0		45
	Up	1		49	**98**	50
After	Dn	132	**95**	7		139
	Up	11		137	**92.6**	148
Recent	Dn	35	**100**	0		35
	Up	0		26	**100**	26

The accuracies are higher using the support vector machine than the decision trees and even hit 100% in three of the eight summary cells. Next, we look at the use of the variables by the models. Table 7 lists the relative importance of the variables in each model. Again, the top five variables are in bold and the highest variable is shaded. The variable with the most impact in determining the direction that oil will move next week in the Before set is the percent change in the value of net imports; for the Crisis data, it moves to the direction of the Cushing spot price; in the After period it switches to the percent change in the Cushing spot price, and in the Recent period, production has over twenty-five percent of the importance. None of the most important single variables are the same in the support vector machines as they were in the decision trees, an indication that the way the models are making their decisions is very different. If we look at these variables in a condensed format, obtained by summing the variables built on the same base variables, we get the results shown in Table 8. Here we have the most important variable in bold and shaded. Net imports came in first in the Before period, Cushing spot price in both the Crisis and After periods, and Production in the Recent period.

Table 7. Relative variable importance.

Variable	SVM Before	SVM Crisis	SVM After	SVM Recent
DirCushingSpot		**0.1323**	0.0149	0.0552
DirNetImports		**0.1158**	0.0369	**0.1235**
DirProduction		0.0905	0.0506	**0.2584**
DirStocksExclSPR	0.0454	**0.0912**	0.0065	0.0493
DirSupply	0.0240	**0.0957**	0.0069	0.0257
PerChgCushingSpot	**0.1055**	0.0737	**0.1737**	0.0614
PerChgNetImp	**0.1875**	0.0692	0.0299	**0.1174**
PerChgProduction	0.0447	**0.0944**	**0.0729**	
PerChgStocksExclSPR		0.0487	**0.1541**	0.0575
PerChgSupply	**0.1263**	0.0707	**0.1000**	**0.0659**
ScldCushingSpot	0.0506	0.0444	**0.1456**	**0.0829**
ScldNetImports	**0.1691**	0.0169	0.0675	0.0352
ScldProduction	**0.1152**	0.0089	0.0438	0.0197
ScldStocksExclSPR	0.0406	0.0158	0.0469	
ScldSupply	0.0911	0.0318	0.0498	0.0479

None of these match the most important base variables in the decision tree models. Like the decision tree models, in both the Crisis and After periods, the most important base variable was the same.

Table 8. Table 6. Sum of Dir, PerChg and Scld for each of the base variables, per period.

Variable	SVM Before	SVM Crisis	SVM After	SVM Recent
CushingSpot	0.1561	**0.2504**	**0.3342**	0.1995
NetImports	**0.3566**	0.2019	0.1343	0.2761
Production	0.1599	0.1938	0.1673	**0.2781**
StocksExclSPR	0.0860	0.1557	0.2075	0.1068
Supply	0.2414	0.1982	0.1567	0.1395

3.3 Both methods

We have seen in the previous sections that the decision tree models and the support vector machine models both do well, but they value and use the variables in different ways. Neither model is always right across all periods. It would be of interest to see whether they are wrong at the same times. Figure 4 shows a graph for each period of when each of the models is wrong. The horizontal axis in each case spans from the beginning to the ending date of the period. We see that some of the time, both models are wrong, however much of the time, these incorrect forecasts occur in different weeks.

Figure 4. Dates when each of the models predicted the wrong direction

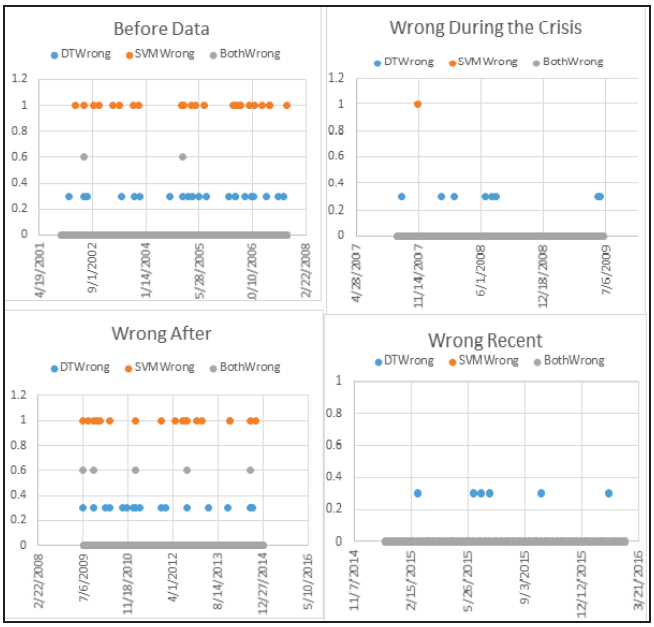


Table 8 summarizes the percent of time that the individual models are correct and are wrong, and, when they agree, the percent of time that they are both correct. We can see that, in the Before period, agree occurs 262 out of 303 weeks. In these 262 weeks, the models are correct 260 out of the 262 time, that is, 99.24% of the time.

During the Crisis period, there is agreement 86 out of 95 weeks. When the models agree, they are correct always. In the After period, the models agree 262 out of 287 weeks and, when such agreement occurs, are correct 98.09% of the time. In the last period, Recent, the models agree 55 out of 61 weeks, and are correct 90.16% of the agreement time.

Thus, in all periods but the last one, we improve our forecasting accuracy by looking only at the weeks when both of the models agree. This gives us an indication that we might be more successful in trading during weeks when the model signals are in agreement.

Table 8. Model agreement.

	DT alone	SVM alone	When Agreed	#Weeks Agreed
Before: 303 weeks				
Correct	93.07%	92.08%	99.24%	260
Wrong	6.93%	7.92%	0.76%	2
Crisis: 95 weeks				
Correct	91.58%	98.95%	100%	86
Wrong	8.42%	1.05%	0%	0
After: 287 weeks				
Correct	94.08%	93.73%	98.09%	257
Wrong	5.92%	6.27%	1.01%	5
Recent: 61 weeks				
Correct	90.16%	100.0%	90.16	55
Wrong	9.84%	0.00%	9.84	0

4 Conclusions

The purpose of this study is to explore two questions: what determines the forecastability of oil prices, and whether two data mining models that "think" very differently can be used to generate a better forecast that either model alone. In contrast to market efficiency that proposes that financial markets rationally collect all relevant fundamental data to accurately determine spot prices without any ability to forecast future prices, this study explores two strategies for understanding the formation of the direction of future prices a week ahead. This study used in-sample data in the analysis. The models help to identify the relative importance of the inputs.

The study considers four regimes and finds that the economic fundamentals play a different role in each regime. In addition, it makes use of two different methodologies to assess the changing role of fundamentals. The high accuracy of the in-sample forecastability indicates that the existing structure among fundamentals was successfully detected by the two methods used and actually by jointly employing these two methods, the overall results improve further.

5 References

[1] Baumeister, C., & Kilian, L. (2014). What central bankers need to know about forecasting oil prices, International Economic Review, 55(3), 869-889.

[2] Baumeister, C., Kilian, L. (2015) Forecasting the Real Price of Oil in a Changing World: A Forecast Combination Approach, Journal of Business & Economic Statistics, 33(3), pp 338-351.

[3] Chen, S. (2014) Forecasting Crude Oil Price Movements with Oil-Sensitive Stocks, Economic Inquiry, 52(2), pp 830-844.

[4] Jammazi, R., Aloui, C. (2012) Crude oil price forecasting: Experimental evidence from wavelet decomposition and neural network modeling, Energy Economics, 34(3), pp 828-841.

[5] Shin, H., Hou, T., Park, K., Park, C., and Choi, S. (2013) Prediction of movement direction in crude oil prices based on semi-supervised learning, Decision Support Systems, 55(1), pp 348-358.

[6] Tsai, C. (2015) How do U.S. Stock Returns Respond Differently to Oil Price Shocks Pre-Crisis, Within the Financial Crisis, and Post-Crisis?, Energy Economics, 50, pp 47-62.

[7] Xiong, T., Bao, Y., Hu, Z (2013) Beyond one-step-ahead forecasting: Evaluation of alternative multi-step-ahead forecasting models for crude oil prices, Energy Economics, 40, pp 405-415.

[8] Noguera, J. (2013) Oil Prices: Breaks and Trends, Energy Economics, 37, pp. 60-67.

[9] Salisu, A., & Fasanya, I (2013) Modelling Oil Price Volatility With Structural Breaks, 52, pp. 554-562.

Inter-correlative Histogram Feature and Dimension Reduction for Content Based Multimedia Retrieval

Vinoda Reddy[1], P.Suresh Varma[2], A.Govardhan[3]
[1]Associate Prof. & Head, CSE Dept., SIT, Gulbarga, Karnataka, India.
[2] Prof.,Dean, Dept of CSE, Adikavi Nannaya University, Rajahmundry, India.
[3]Prof. & Principal, JNTUH College of Engineering Hyderabad, Telangana, India.

Abstract: **Information retrieval based on motion descriptions in multimedia application has gained lot of importance and interest in the recent past. Content Based Multimedia Retrieval (CBMR) system retrieves the multimedia data based on the content of given multimedia query. That is, for a given motion query, relevant multimedia are to be retrieved. Here, an approach of human action detection and localization in a multimedia dataset and an improved histogram based approach for Multi-Instant action detection named as Multi-Instant Histogram (MI-HIST) is proposed. The proposed approach enhances the disadvantages like processing noise and system overhead with respect to the representation and retrieval performance. In the representation of descriptive features, a large feature count is observed, which results in the processing overhead. To reduce these descriptive features, this paper also presents a multi-linear kernel (MLK) technique for dimension reduction approach to feature reduction based on feature relations. The experiments have been conducted on the benchmark datasets. The experimental results show that the processing resource overhead and retrieval efficiency are improved with the present approach.**

Keywords: CBMR, Histogram features, PCA, LDA, Kernel, Accuracy, and Computation Time.

I.INTRODUCTION

Multimedia Retrieval is one of the most important and fastest growing research areas in the field of multimedia technology. Large collections of scientific, artistic and commercial data comprising image, text, audio and video abound in the present information based society. There is a need for an effective and precise method of assisting users to search, browse and interact with these collections. As per the present trend, people use not only pure images for daily purposes, but also video is a popular media for recording TV, diaries etc. As a consequence, effective and efficient methodsfor searching with large databases are needed. This motivates the need for using Content Based Multimedia Retrieval (CBMR) [1] System for images or videos which allowusers to search images or particular image frames according to their preferences. However, today's multimedia retrieval systems are more inclined towards image retrieval model and less focus is made on the retrieval of video content.

Video information is processed in real time applications such as surveillance, film making, home monitoring, CCTV, monitoring etc. with higher storage resources and new capturing units. In such applications, video information could result in more informative than their corresponding images, retrievals of information over such system that are limited. Current multimedia databases such as YouTube use a text based searching mechanism to search video content. A video annotation based retrieval [2, 3] and a video summarization based approach [4] to obtain keyword based action retrieval are proposed earlier. However, the retrieval performance is purely dependent on the tagging factor for applications. It is very difficult to extract video based on the content, such as action in a video samples. A motion based coding following energy and a history image[5] was also used as an action detection model. A local representation of the extracted spatio-temporal interest points (STIPs) [6, 7] from an action is developed earlier. The local representation is highly robust to statistical representation of action dataset [8, 9]. In the action detection model, Harris detector [10] and 3-D SIFT [11, 12] are used for action detection process. The SIFT operator performs a max/min searching over the difference of Gaussian (DoG) function. A combined format of histogram oriented gradient and histogram optical flow (HoG-HoF) [13] was also developed previously. The histogram of gradient is observed to be an effectively applied approach for action detection model. A 3D-HoG [14] is developed earlier with a set of descriptive approaches to obtain effective action model. Further, a histogram based coding [15] for video retrieval is developed for the optimization of action detection model. The system uses the localized temporal histogram features to detect an action model from a video dataset. However, as observed the Noise factor impacting the histogram representation is considered over two successive frames only. The global distribution of noise distribution over the video frames is not analyzed in representing the Histogram feature. This factor affects in the retrieval accuracy and increases the resource overhead in multimedia retrieval approach.Feature sets are very important in the approach of multimedia retrieval. Xin Geng et al. [16][17] have proposed the Representation pattern Subspace method for multimedia retrieval based on pattern recognition. Its idea is to model the representation pattern, which is defined as a sequence of personal representation pattern videos by learning a representative sub-space from EM-like (expectation maximization) iterative learning Principle Component Analysis (PCA).

Frequency domain analysis is the most popular method for extracting video features in video processing and pattern recognition. Guodong Guo et al. [20] have investigated the biologically inspired features (BIF) for pattern recognition from given videos. Unlike the previous works [18][19], Guoetal. simulated the human visual process based on bio-inspired models [21] by applying Gabor filters. A Gabor filter is a linear filter used in video processing for edge detection. Frequency and orientation representations of Gabor filters are similar to that of human visual system and have been found to be appropriate particularly for textural representation and discrimination. Though PCA based coding is observed to be applied over multiple applications, the feature representation is higher and the approach of selective operation minimizes the feature selection accuracy. In previous work multi instance histogram features from the MI-HIST [22] are processed for dimensionality reduction through MLK-DR presented[23]. In this paper, an approach for inter frame computation of histogram features and selection is proposed which extracts sufficient features for retrieval to overcome the representation of Histogram based coding considering noise factor. Further, this paper also proposes a dimensionality reduction technique which reduces the computational overhead of retrieval system. The content of this paper is organized into 6 sections. Section 1 presents the objectives and related work on information retrieval in content based multimedia retrieval system. Section 2 defines the system model and section 3 defines the approach of histogram for inter frame localization and representation. Section 4 outlines the dimensionality reduction technique. Section 5 outlines the obtained experimental results for the developed approach. The conclusionsof contributions are presented in section 6.

II. MULTIMEDIA RETRIEVAL SYSTEM

The general multimedia retrieval system consists of three stages: training, testing and classification. The training stage trains the various multimedia videos of benchmark datasets along with their features. Testing stage extracts the features of a query sample and gives it to classifier. The classifier classifies the given query sample by comparing it with the trained samples. The developed system for the proposed multimedia retrieval is given below:

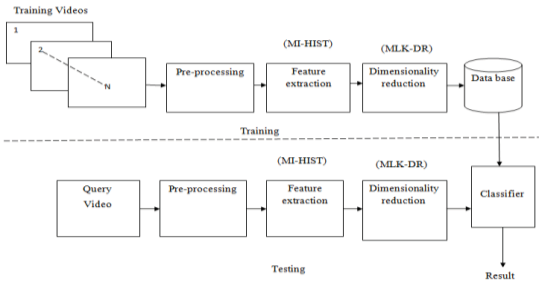


Figure 1. Model of proposed multimedia retrieval system

The developed approach is processed into two phases of execution of training and testing process. In training process, set of video samples of different actions like running, walking, jumping and sitting is taken andthese are processed for feature selection through the proposed Multi Instance-Histogram (MI-HIST) approach. Further, these features are then processed for Multi Linear Kernel Dimensionality Reduction (MLK-DR)technique. Finally, the reduced feature set is processed for classification using a SVM classifier.

III. MULTI INSTANCE-HISTOGRAM (MI-HIST)

An approach was proposed for information retrieval based on the actions of the objects through their histogram features [8]. This approach defines a temporal and spatial localization of an action model based on Histogram mapping. However, the noise effect leads to the reduction of retrieval performance. An inter frame correlation error for a set of time frames are considered to eliminate the impact of noise. Considering a set of Histogram for k class (H_k), for a given video dataset of i=1 to k,

$$H_i(k) = [H_i(kN), H_i(kN-1), \ldots\ldots, H_i(kN-M+1)\ ] \quad (1)$$

Where, $H_i$is the Histogram for a video frame, N is number of frame and $M$is the dataset samples. A frame error is computed to evaluate the noise effect in the temporal framesas defined by,

$$e_{i,H}(k) = H_{i,t}(k) - H_{i,t+1}(k) \qquad (2)$$

This error defines the difference in two frame components and the histogram errors with lower values $\min\left(e_{i,H}(k)\right)$are considered as feature element. However, this error deviatesmuch when observed over a period of frame observation and could be effective due to noise effect. Hence, the intersection histogram would be mode concentric with noise parameterin such coding. A Histogram bin selection computed over a time series is proposed to eliminate this problem and to improve the feature selection more accurately. In this approach, selection of the histogram bins is made rather than taking the whole histogram from single frame information. The histogram bins are initially normalized using a random weight factor to derive the bin selection.

$$H_i(k) = H_i(k)\, w(k) \qquad (3)$$

Where$w(k) = [w_0(k),\ w_1(k), \ldots\ldots, w_{M-1}(k)]^T$isthe allocated weight factor for each frame? Then the estimated error is then defined as:

$$e_{i,H}(k) = H_{i,t}(k) - H_i(k)w \qquad (4)$$

The error is recursively computed over the total frames (i=1,…, N) and the initial error is recorded as $e_{i,H,init}$.A weight factor is then updated as:

$$w(k+1) = w(k) + \mu \sum_{i=0}^{N-1} \frac{H_i^T(k)}{\|H_i(k)\|^2} e_{i,H,init}(k) \qquad (5)$$

Where μ is the updation step size, with an error updation factor. The objective of this computation is to select the bins satisfying the $\min(e_{i,H}(k))$ condition. A joint adjacent weight difference is computed to optimize the recursion overhead and is defined as:

$$\widetilde{w}(k) = w^o - w(k) \qquad (6)$$

Where, w^o is the initial weight issued.The weight updation is then defined as,

$$\widetilde{w}(k+1) = \widetilde{w}(k) - \mu \sum_{i=0}^{N-1} \frac{H_i^T(k)}{\|uH_i(k)\|^2} e_{i,H}(k\ \ ) \quad (7)$$

The deviation in the bin variationof the histogram is then integrated over a period of 0 to N defined by,

$$\mathrm{E}(H_{i,N}) = \int_0^N \mu \sum_{i=0}^{N-1} \left(2E\left[\frac{H_{i,n}(k)\tilde{w}(k)e_{i,NH}(k)}{\|H_{i,N}(k)\|^2}\right] - \mu E\left[\frac{e_{i,NH}^2(k)}{\|H_{i,N}(k)\|^2}\right]\right) \quad (8)$$

Wherein integrating the estimate, over 'n' observation period accumulates the estimation for 'n' inter frame errors. For each frame with minimum estimate error is then selected as the selected histogram bin and an intersection bin is then derived as:

$$s(H_q, H_t) = \sum_{i=1}^{k} \left(\frac{\min(H_q - H_t)}{H_n}\right) \quad (9)$$

Where $H_n$is the normalized histogram for the entire dataset.

The obtained multi instance histogram features are processedfor dimensionality reduction. The dimensionality reduction method reduces the dimensions of obtained feature setso that the computation overhead can be minimized.

IV. MULTI-LINEAR KERNEL DIMENSIONALITY REDUCTION (MLK-DR)

The main aim of any dimensionality reduction approach is to minimize the number of features to be processed. Less the number of features, less will be the computational overhead and less computation time. So the obtained multi instance histogram features from the MI-HIST[22] are processed for dimensionality reduction through MLK-DR presented [23]. MLK-DR stands for Multi Linear Kernel Dimensionality Reduction. It is a multi linear subspace learning method that extracts features directly from multi-dimensional objects. The MLK-DR is an extension to the conventional PCA [16], which operates linearly whereas MLK-DR operates multi-linearly. The PCA needs to reshape the multidimensional object into the vector, whereas MLK-DR operates directly on multidimensional object through two-mode processing. In this paper, the histogram features obtained through MI-HIST are given as inputto MLK-DR. The operation for the PCA is defined as: for a given data set of N-by-N dataset samples, I(x, y) as a vector of dimension N^2, so that the sample can be thought of as a point in N^2-dimensional space. A database of M samples can therefore map to a collection of points in this high dimensional "dataset space" as $\Gamma_1,\Gamma_2,\Gamma_3, \ldots,\Gamma_M$. The average dataset of the sample set is defined as:

$$\Psi = \frac{1}{M}\sum_{n=1}^{M} \Gamma_n \quad (10)$$

Each dataset is mean normalized and is represented as deviations from the average dataset by$\Phi_t = \Gamma_t - \Psi$. The covariance matrix is defined as the expected value of $\Phi\Phi^T$and can be calculated by the equation:

$$C = \frac{1}{M}\sum_{n=1}^{M} \Phi_{n-1}\Phi_n^T \quad (11)$$

Given the covariance matrix *C*, one can now proceed with determining the eigenvectors *u* and eigenvalues λ of *C* in order to obtain the optimal set of principal components and a set of Eigen datasets that characterizes the variations between dataset samples. Consider an eigenvector $u_i$of *C* satisfying the equation

$$C_{u_i} = \lambda_i u_i \quad (12)$$

$$u_i^T C_{u_i} = \lambda_i u_i^T u_j \quad (13)$$

The eigenvectors are orthogonal and normalized. Hence,

$$u_i^T u_j = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \quad (14)$$

Combining Eq. (11) and (14), Eq. (13) thus become

$$\lambda_i = \frac{1}{M}\sum_{n=1}^{M} var(u_i \Gamma_n^T) \quad (15)$$

Eq. (15) shows that the eigenvalue corresponding to the i_{th} eigenvector represents the variance of the representative dataset sample. By selecting the eigenvectors with the largest corresponding eigenvalues as the basis vector, the set of dominant vectors that express the greatest variance are being selected. The PCA algorithm applied for dimensionality reduction reduces the dimensions of obtained feature set only by considering internal variations in that particular sample only. However, the inter class variations, i.e., the features of other frames in a sequence are not considered. The PCA operates in one dimensional mode, whereas MLK-DR operates along multi-dimensional mode. For a given feature space of a single class, PCA evaluates the principal components individually, whereas MLK-DR evaluates by considering the feature space of remaining classes also. The pseudo-code for MLK-DR is described as follows:

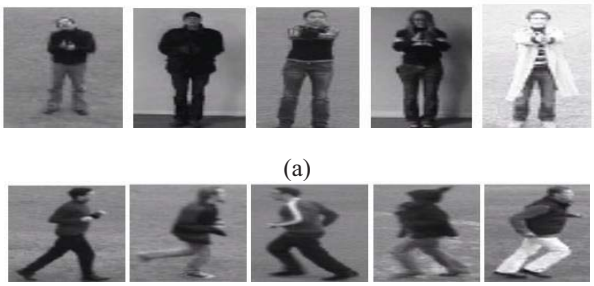
***Pseudo Code**:*
***Input:** Features of large size*
***Output:** Feature set with less size*
Step 1: take the whole feature set having MxN dimensional space
Step 2: compute the mean along each 'n' dimensions
Step 3: obtain a new matrix by subtracting the mean from all values of dataset.
Step 4: evaluate a covariance matrix
Step 5: compute Histogram vectors and the corresponding Histogram values
Step 6: sort the Histogram vectors by decreasing the Histogram values and choose k Histogram vectors with largest Histogram values from n x k dimensional matrix W_1.
Step 7: perform the same operation of step7 for each class of feature set and find out some more Histogram values those having effect on the retrieval accuracy.
Step 8: finally form a new projection matrix by considering inter class Histogram values and also intra class Histogram values.
Step 9: form a dimensionally reduced subspace by multiplying the projection matrix with original values.

Then, the feature vectors are compared with the database using SVM classifier and classify the multimedia video as to which class it matches.

V. SIMULATION RESULTS

The proposed system is developed over Matlab tool and tested over Weizmann benchmark dataset and KTH

dataset [7]. The KTH video database contains six types of human actions (walking, jogging, running, boxing, hand Waving and hand clapping) performed several times by 25 subjects in four (4) different scenarios: outdoors s1, outdoors with scale variation s2, outdoors with different clothes s3 and indoors s4 as illustrated below. Currently the database contains 2391 sequences. All sequences are taken over the homogeneous backgrounds with a static camera with a frame rate of 25fps. The sequences are down sampled to the spatial resolution of 160x120 pixels and have a length of four seconds in average.A comparative analysis of PCA based feature dimension reduction is compared with the proposed MLK-DR to evaluate the performance of the proposed approach. The simulation is performed over KTHdataset for which MI-HIST features are computed. The test dataset is shown in Figure 2.



(a)

(b)

(c)

(d)

Figure 2: Dataset with (a) Handclapping (H), (b) Jogging (J), (c) Running (R) and (d) Walking (W) sample

The samples are captured with 180 x144 resolution by using a static camera with a homogenous outdoor background.The processing test sample that has a running action is illustrated in Figure 3.

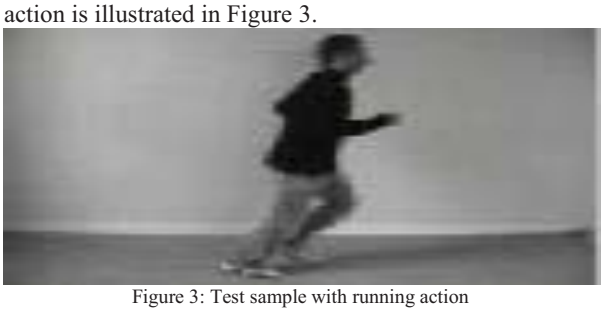


Figure 3: Test sample with running action

The obtained features of the MI-HIST and MLK-DR are represented in Table 1.

Table 1. Feature table of various approaches

Class	Feature Extraction			Dimensionality Reduction		
	HoG[14]	HIST[15]	MI-HIST [22]	PCA[16]	LDA[19]	MLK-DR [23]
C1(H)	43246	21452	14712	6210	5342	4640
C2(J)	72478	52984	37862	14980	14320	12546
C3(R)	61146	44102	21476	8480	7742	6320
C4(W)	44528	35284	18438	7245	6242	5540

Table 1 illustrates the obtained feature count for both feature extraction methods and dimensionality reduction methods. The feature extraction methods extract only features. After feature extraction, the entire features obtained are trained as well as tested through the classifier, whereas in dimensionality reduction methods, the dimensions of obtained feature set are reduced first and then only they are processed for training as well as testing. Hence, the feature count of dimensionality reduction methods is less. In Table 1, the obtained features for PCA, LDA and for MLK-DR are compared with HOG, HIST and MI-HIST. The obtained feature count of various approaches at feature extraction stage and at dimensionality reduction stage is shown in Figure 4.

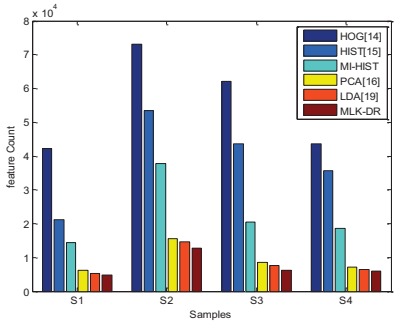


Figure 4. Feature count for four actions

The complete system is simulated in two phases: training phase and classification phase. The training phase establishes a database with a set of features using feature extraction techniques. The classification phase performs the classification for a given query input. The time taken for training is termed as training time and the time taken for classification is termed as classification time. These two timings vary from approach to approach. The complete analysis of the time consumption for various approaches is represented in Figure 5.

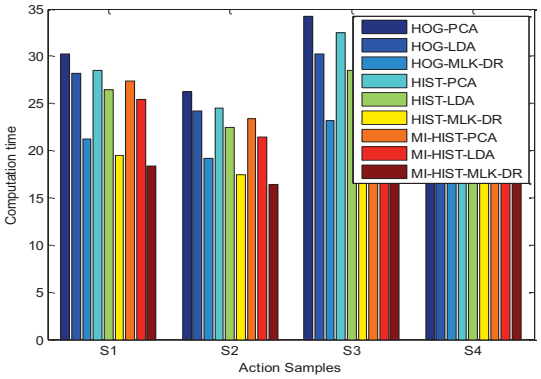


Figure 5. Computation time

Figure 5 illustrates the details of computation time in seconds. The training phase has a large number of samples to process. Hence, the computation time for training is more when compared to that of classification. The details of the computation time are represented for the combination of feature extraction methods with dimensionality reduction methods in Figure 5. It can be observed from Figure 5 that, for each class, the combination of MI-HIST with MLK-DR has less computation time when it is compared with remaining combinations. The following parameters are used to evaluate the performance of the developed approach:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \qquad (16)$$

Where,
TP =True Positive (correctly identified)
FP= False Positive (incorrectly identified)
TN=True Negative (correctly rejected)
FN=False negative (incorrectly rejected)

For the above given simulation model, there are totally four classes and each class has five subjectsthat are processedfor training. In the testing phase, a query sample with running action is selected and is given for SVM classifier. The SVM classifier compares the sample features of given query with database features.
To show the enhancement of proposed approach and also to compare the proposed approach with previous approaches, precision along with accuracy is evaluated as:

$$Precision = \frac{TP}{TP+FP} \qquad (17)$$

The obtained accuracy and precision details are illustrated in Figure 6 and Figure 7 respectively.

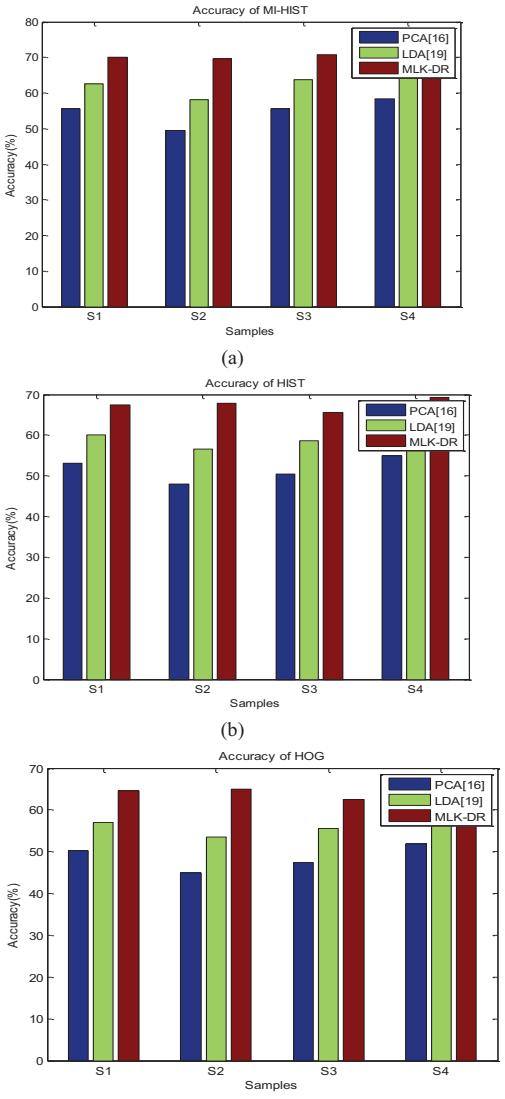


Figure.6 Accuracy of four action samples with dimensionality reduction methods, PCA, LDA and MLK-DR (a) MI-HIST, (b) HIST [15], (c) HOG [14]

Figure 6 illustrates the accuracy details of the proposed work. The proposed work combines the feature extraction with dimensionality reduction approaches. Compared with alone, the combination will have more accuracy. The proposed work combined the HOG, HIST and MI-HIST with PCA, LDA and MLK-DR. the individual details of MI-HIST, HIST and HOG are shown in Figure 6 (a), (b) and (c) respectively. From Figure 6, it can be observed that for the combination of MI-HIST with MLK-DR having more accuracy.

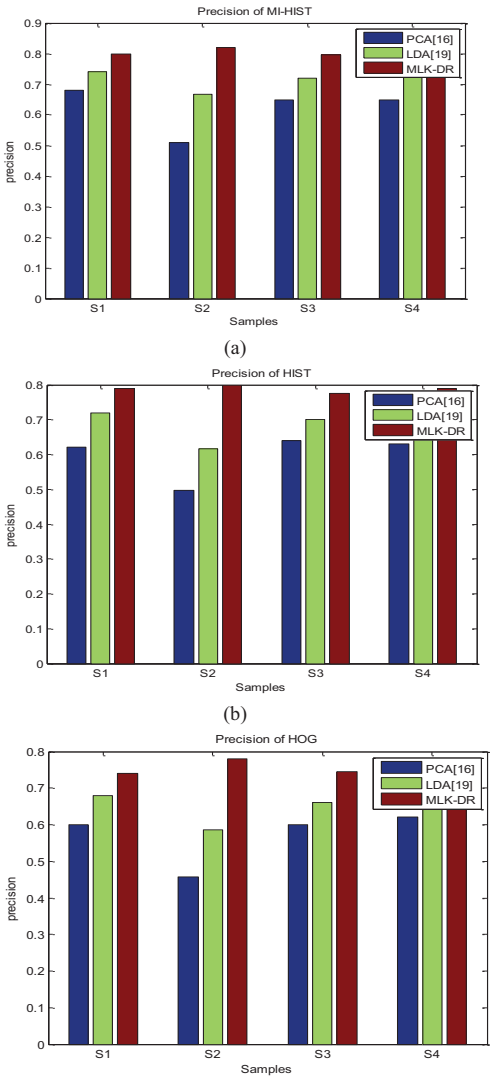


Figure.7 Precision of four action samples with dimensionality reduction methods, PCA, LDA and MLK-DR (a)MI-HIST, (b)HIST [15], (c)HOG[14]

Figure 7 illustrates the precision details of the proposed work. The proposed work combines the feature extraction with dimensionality reduction approaches. It can be observed from Figure 7 that the combination of MI-HIST with MLK-DR gives more precision than these individual techniques.

VI. CONCLUSION

A new coding approach for histogram based action model detection is proposed. A process of inter frame histogram coding for feature selection and its representation is developed. The respective dominant features are obtained with the application of Histogram on video. Histogram features are extracted after obtaining all possible variations from the frame information. In conventional approach, the Histogram features are directly extracted from multimedia video thus finding the dominating features become complex. The proposed work applies dimensionality reduction method in a multi-nature and it reduces the dimensions of feature set by considering the intra class feature and also inter class features, whereas the conventional approach reduces the dimensions by considering intra class featuresonly.The retrieval performance is improved with the optimal selection of histogram features by the application of feature extraction and dimensionality reduction method. The results of the extensive simulation have shown the effectiveness of proposed work with improved performance.

REFERENCES

[1] Del Bimbo, A. (1998). "A Perspective View on Visual Information Retrieval Systems", *Proceedings of the IEEE Workshop on Content-Based Access of Image and VideoLibraries*, Santa Barbara, California.

[2] M. Wang, X.-S. Hua, R. Hong, J. Tang, G.-J. Qi, and Y. Song, "Unified video annotation via multigraph learning", *IEEE Transaction on Circuits and System for Video Technology*, Vol. 19, No. 5, May 2009, pp. 733–746.

[3] M. Wang, X.-S. Hua, J. Tang, and R. Hong, "Beyond distance measurement: Constructing neighborhood similarity for video annotation," *IEEE Transaction on Multimedia*, Vol. 11, No. 3, Apr. 2009, pp. 465–476.

[4] M. Wang, R. Hong, G. Li, Z.-J. Zha, S. Yan, and T.-S. Chua, "Event driven web video summarization by tag localization and key-shot identification", *IEEE Transactions on Multimedia,* Vol. 14, No. 4, Aug. 2012, pp. 975–985.

[5] J. W. Davis and A. F. Bobick, "The representation and recognition of human movement using temporal templates,"*IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9-28, 1997.

[6] I. Laptev, "On space-time interest points," *Journal of Computer Vision*, Vol. 64, No.2, 2005 pp. 107–123.

[7] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition via sparse spatio-temporal features," *Joint International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, IEEE,* 2005 pp. 65–72.

[8] T. H. Thi, J. Zhang, L. Cheng, L. Wang, and S. Satoh, "Human action recognition and localization in video using structured learning of local space time features," *IEEE International Conference on Advance Video Signal Based Surveillance,* 2010, pp. 204–211.

[9] M. Ryoo and J. Aggarwal, "Spatio-temporal relationship match: Video structure comparison for recognition of complex human activities," *IEEE International Conference Computer Vision*, 2009, pp. 1593–1600.

[10]K. Mikolajczyk and C. Schmid, "An affine invariant interest point detector," *European Conference Computer Vision,* 2002, pp. 128–142.

[11]D. G. Lowe, "Distinctive image features from scale-invariant key points," *International Journal Computer Vision,* Vol. 60, No. 2, Nov. 2004, pp. 91–110.

[12]P. Scovanner, S. Ali, and M. Shah, "A 3-dimensional SIFT descriptor and its application to action recognition," *ACM Multimedia,* 2007, pp. 357–360.

[13]I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," *IEEEConference Computer Vision Pattern Recognition,* 2008, pp.1–8.

[14]A. Kl¨aser,M. Marszałek, and C. Schmid, "A spatio-temporal descriptor based on 3-D-gradients," *Brit. Mach. Vision Conf.*, 2008, pp. 995–1004.

[15]Ling Shao, Simon Jones, and Xuelong Li," Efficient Search and Localization of Human Actions in Video Databases",

IEEE Transactions on Circuits and Systems for Video Technology, Vol. 24, No. 3, March 2014, pp. 504-512.

[16]Geng X, Zhou Z-H, Zhang Y, Li G, Dai H. "Learning from multimedia representation patterns for automatic age recognition," *In ACM Conf. on Multimedia*, pp. 307–316, 2006.

[17]GuodongGuo, Guowang Mu, Yun Fu, Charles Dyer, "A Study on Automatic Age Estimation using a Large Database", Computer Vision, *12th International Conference*, pp. 1986 – 1991, IEEE, 2009.

[18]Guo G, Fu Y, Dyer, C.R., Huang, T.S. "Video-Based Human Pattern Recognition by Manifold Learning and Locally Adjusted Robust Regression," *IEEE Transactions on Video Processing*, Vol. 17, pp.1178-1188, 2008.

[19]Guo G, Fu Y, Huang T.S. and Dyer, C.R. "Locally Adjusted Robust Regression for Human Age Recognition," *IEEE Workshop on Applications of Computer Vision*, pages 1-6, 2008.

[20]GuodongGuo, Guowang Mu, Yun Fu, Thomas S. Huang, "Human Age Estimation Using Bio-inspired Features," *Conference on Computer Vision and Pattern Recognition*, pp. 112 – 119, 2009.

[21]Serre T, Wolf L, Bileschi S, Riesenhuber M andPoggio T. "Robust Object Recognition with Cortex-Like Mechanisms," *IEEE Trans. on PAMI*, 29(3): 411–426, 2007.

[22]Vinoda Reddy, P.SureshVarma and A.Govardhan, "Recurrent Energy Coding For Content Based Multimedia Retrieval System", *International Journal of Multimedia and User Design & User Experience*, Vol.24, 2015.

[23]Vinoda Reddy, P.SureshVarma and A.Govardhan, "Multilinear Kernel Mapping For Feature Dimension Reduction in Content Based Multimedia Retrieval System", *The International Journal of Multimedia & Its Applications (IJMA),* Vol.8, No.2, 2016.

Detecting Mass Emergency Events on Social Media: One Classification Problem or Many?

V. Pekar[1], **J. Binner**[1], **H. Najafi**[2]
[1]Business School, University of Birmingham, Birmingham, United Kingdom
[2]Computer Science and Information Systems, University of Wisconsin, River Falls, WI, USA,

Abstract *Social media proves to be a major source of timely information during mass emergencies. A considerable amount of recent research has aimed at developing methods to detect social media messages that report such disasters at early stages. In contrast to previous work, the goal of this paper is to identify messages relating to a very broad range of possible emergencies including technological and natural disasters. The challenge of this task is data heterogeneity: messages relating to different types of disasters tend to have different feature distributions. This makes it harder to learn the classification problem; a classifier trained on certain emergency types tends to perform poorly when tested on some other types of disasters. To counteract the negative effects of data heterogeneity, we present two novel methods. The first is an ensemble method, which combines multiple classifiers specific to each emergency type to classify previously unseen texts, and the second is a semi-supervised generic classification method which uses a large collection of unlabeled messages to acquire additional training data.*

Keywords: text classification, semi-supervised learning, social media analysis, disaster management

1 Introduction

Social media data offer a very promising way forward as a mechanism for facilitating the work of first responders in dealing with mass emergency events. During a crisis such as a natural disaster or a terrorist attack, social media has become a primary source of information, publishing eyewitness reports on the events in real-time. Information systems that identify and collate such eyewitness reports can provide critical situation awareness to increase the efficiency and capabilities of the emergency services, making them better equipped to detect disasters at early stages, monitor their development and tackle their consequences in the recovery operations.

The potential of social media analysis for mass emergency management has attracted many researchers in the fields of Data Mining over the past several years. Previous work has primarily focused on detecting emergency-related tweets using text classification. Limiting the problem to a single disaster type such as earthquakes or tornados has been shown to achieve high accuracy of classification (e.g., [4, 8, 9, 10]). However, because mass emergency events can differ a lot in terms of their causes, temporal and geographical spread, impacted targets and the nature of damage, it would be much more practical to have a classification method that can cover a wide range of possible disasters. This will give first responders and emergency services personnel confidence that disasters with some previously unseen characteristics would be successfully recognized by the alerting system.

This paper is concerned with the task of recognizing mass emergencies unspecified for a particular type, which could include both natural disasters such as earthquakes, floods and storms, as well as man-made ones such as explosions, collisions and shootings. This is a non-trivial classification problem, as the data is non-homogeneous: the classifier is trained and evaluated on data covering different emergency types; each characterized by its own vocabulary and correspondingly different feature distributions. We examine two possibilities to counter the problem of heterogeneous data. The first approach views this task as multiple classification problems, training one classifier for each type of known disasters; an ensemble of the classifiers is then used to classify test messages that can possibly come from an unknown disaster type. The second approach treats the task as a single classification problem: in order to better capture commonalities that exist between different types of disasters, it uses a co-training method, which obtains additional training data from a large collection of unlabeled messages. Thus our main contributions are the novel methods that are specifically suited to the task of detecting emergency events that were unseen at the training stage and their comparative evaluation.

The paper is organized as follows. In the next section we outline previous work on detecting mass emergencies using machine learning text classifiers. In Section 3 we describe the proposed ensemble classification method, and in Section 4 the co-training method. Sections 5 and 6 present the experimental setup, the results of the experiments and their discussion. Section 7 concludes and offers suggestions for future research.

2 Related work

There is a considerable body of work on detection of new events in a stream of text messages, where the type of the event of interest is not known in advance, and some of these approaches were applied to detecting mass emergency events. Such methods primarily rely on detecting "bursty" keywords [13], i.e. keywords whose frequency increases sharply within

a short time window. However, bursty keywords are known to be related not only to events, but also non-events such as "viral" content. To separate them, Becker et al. [2] used a domain-independent text classifier, before applying keyword burstiness techniques.

Domain-specific methods generally have a greater accuracy than domain-independent ones, and previous work specifically on mass emergency detection was concerned with developing domain text classifiers based on machine learning. The classifiers aim to solve a binary classification problem, operating on features extracted from the entire message. Most of this work was concerned with specific types of disasters such as earthquakes [18, 22, 23], tornados [9, 11], and landslides [14].

Verma et al. [21] conducted experiments on how well a classifier trained on one type of emergency would perform on messages representing a different emergency type. They ran all pairwise comparisons between four datasets, which represented two flood events, one earthquake and one wildfire, and found that testing on an emergency type other than the one used for training results in worse classification accuracy; the F-measure ranging between 29 and 83 depending on a specific pair. Ashktorab et al. [1] trained one generic classifier on data from twelve different emergency events, achieving the F-measure between 50 and 65 depending on the learning method; the evaluation was done however by randomly splitting all the data into a test and train sets, i.e., the train and test data contained data representing different disasters in similar proportions. Pekar et al. [17] showed that if a classifier is trained on some disaster types, but evaluated on others, the performance of the classifier drops by up to 70%, when compared with training and testing on the same set of disasters.

3 Ensemble classification

Ensemble or committee-based classification methods aim to leverage advantages of different models trained on the same dataset, in order to improve on performance of individual models [6]. The different models in the ensemble can be learned using different subsets of the training data, different classification parameters of the same learning algorithm, or using different learning algorithms. For a review of ensemble methods applied to text classification, see e.g. [7].

In the context of detecting disaster-related text messages, we create a classifier ensemble through dividing the training instances by disaster type. We then trained one classifier specific to each type, using the same learning algorithm. Each of the classifiers is thus expected to be more effective at classifying just its own disaster type, than a classifier trained on other types or a generic classifier. Test instances representing an unknown disaster would then be classified more effectively by some classifiers than others. This is because the unknown disaster will be more similar to some of the disaster types observed during training than others.

Majority vote among classifiers is the simplest way to derive the eventual class label for the test instance, but in the case of highly heterogeneous data the majority class will seldom be the correct one. Our initial experiments showed that the negative class almost always got the majority vote. Therefore in our implementation the test instance is given the class label of the classifier that assigned it with the highest confidence.

Input:

Training documents D_{train}

Testing documents D_{test}

Emergency types E

Class labels $Y = \{True, False\}$

Training phase:

For each e in E:

Train classifier c_e on a subset of D_{train} each of which belongs to e

Testing phase:

For each d in D_{test}:

For each c_e in $C = \{c_1, c_2, \ldots c_{|E|}\}$:

Obtain label y_e and classifier confidence score s_e

Assign y_i *to* d, such that s_i is the maximum value in $\{s_{e1}, s_{e2}, \ldots, s_{|E|}\}$

Algorithm 1. Ensemble classification method.

4 Co-Training

Co-training [3] is a semi-supervised learning technique that is aimed to overcome the problem of insufficient training data, if large amounts of unlabeled data are available. It is also commonly used for domain adaptation of a classifier, when train data available for one domain is used to obtain train data for a different domain, thus tuning the classifier to perform more effectively on the new domain. The technique was previously used for domain adaptation for various NLP tasks, including dependency parsing [20], co-reference resolution [15], and sentiment classification of texts [5].

The general co-training algorithm starts with choosing two different "views" on the data, which are disjoint subsets of the entire set of classification features. The subsets are created to satisfy two conditions: (1) each view must be able to learn the classification problem with sufficient accuracy and (2) the views must be conditionally independent of each other. Two classifiers are trained using each view on available labeled training data. They are then used to classify unlabeled data. Instances which either classifier labeled with high confidence are added to the train set, thus the classifiers help each other, adding one's most confident classifications into the other's train set. The classifiers are then re-trained on the new dataset and re-applied to the unlabeled data. These steps are repeated until a certain stopping criterion is reached. After that, the

main classifier is trained on the augmented train set and evaluated on the test data.

In our implementation, feature subsets are created as follows. The first one consists of unigrams and bigrams, i.e. lexical features extracted from the cleaned version of the message. The second one includes grammatical features (part-of-speech tags) and features extracted from Twitter metadata (hashtags, mentions, presence of URLs, etc.). The two subsets were found to produce models of similar classification accuracy (for details of the features used and experiments with feature subsets, see Section 6). Unlabeled instances are added to the training set in such a way as to preserve the ratio of positive and negative instances that was found in the original train data. As a stopping criterion, we used the maximum number of automatically added training instances, which we set at 25% of the original training set.

Input:
Labeled documents L
Unlabeled documents U
Data views V
Classifier confidence threshold t
Desired size of labelled documents m

Initialize:
Augmented labeled set $L' \leftarrow L$

Loop:
While $|L'| < m$ and $|U| > 0$:
    for v in V:
        Train classifier c_v on L' using features from v
    for u *in* U:
        for c_v in C:
            Classify u with c_v, obtaining class label y_v and confidence score s_v
        Select y_i from $\{y_1, y_2, \ldots, y_{|V|}\}$ such that s_i is the maximum in $\{s_1, s_2, \ldots, s_{|V|}\}$
        if $s_i > t$:
            Assign y_i to u and add u to L'
            Remove u from U

Output:
Augmented labeled set L'

Algorithm 2. Acquisition of labeled data in the co-training algorithm.

5 Experimental setup

5.1 Labeled data

In the experiments we use the labeled part of the CrisisLexT26 dataset [16], which includes tweets on twenty six mass emergencies that occurred between 2012 and 2013. The types of emergencies are very diverse and range from terrorist attacks and train derailment to floods and hurricanes. Some examples are Colorado wildfires in 2012, Venezuela refinery explosion in 2012, and Boston bombings in 2013. The dataset was created by first retrieving tweets based on a set of search terms relating to mass emergencies, and thus is representative of data that is likely to be found in real-world use cases after initial keyword-based filtering.

The evaluation included three classification tasks, which are of different practical value for emergency responders and at the same time differ in terms of the difficulty of the classification problem:

i. Relatedness: separating messages related to a mass emergency from unrelated ones,

ii. Informativeness: separating informative messages (whether the message contributes to better understanding of the crisis situation) from uninformative ones (refers to the crises but involves sympathy, jokes, etc.),

iii. Eyewitnesses: detecting eyewitness accounts of mass emergencies (first-hand descriptions of the events).

Figure 1 shows examples of positive and negative messages for the three tasks (examples taken from Olteanu et al. [16]).

Table 1 describes the size of the positive and negative classes in the three classification tasks in the CrisisLexT26 dataset.

	Positive	Negative
Relatedness	24581	2863
Informativeness	16849	7732
Eyewitnesses	2193	22396

Table 1. The sizes of the positive and negative classes in the classification tasks.

	Positive	Negative
Relatedness	RT @NWSBoulder Significant flooding at the Justice Center in #boulderflood	#COstorm you are a funny guy lol
Informativeness	Flash floods wash away homes, kill at least one near Boulder via @NBCnews	Pray for Boulder, Colorado #boulderflood
Eyewitnesses	Outside sounds like it is going to shatter my bedroom windows any sec now #bigwet #qld	RT @RedCrossAU: Everyone affected by #qldfloods, let people know you're safe: http://t.co/..

Figure 1. Examples of messages belonging to positive and negative classes of the three classification tasks.

5.2 Unlabeled data

To obtain unlabeled data for co-training experiments, we created 24 search terms describing different types of mass emergencies: *avalanche, blizzard, cyclone, earthquake, flood, landslide, heat wave, eruption, storm, tornado, tsunami, wildfire, bushfire, crash, explosion, collision, disaster, shooting, accident, capsize, sank, stampede, collapse, massacre*. Submitting the search terms to the Twitter Search API we continuously retrieved tweets that were published between February 23, 2016 and March 08, 2016, obtaining 2,479,079 tweets in total.

5.3 Preprocessing

We apply the following preprocessing steps to the data, which are commonly used for Twitter messages before performing text classification on them in order to reduce the amount of noisy features (see, e.g.,[12]):

- **Text normalization**. Before processing the text of the message with a PoS tagger, the text was normalized: mentions (e.g., @username) and URLs removed; sequences of hashtags at the start and end of the message removed; hashtags appearing in the middle of the text were kept, but the hash symbol removed from the hashtags; long non-alphanumeric symbol sequences, which tend to be emotions, were removed; word tokens consisting of digits were replaced with a unique tag.
- **Part-of-speech tagging**. The normalized text was tagged with the PoS tagger in the Pattern library [19].
- **Stopword removal**. The usual stoplist was used to remove stopwords.
- **Additional metadata**. The CrisisLexT26 data contains the Twitter id of the message, its raw content, and its timestamp. We retrieve via Twitter Search API additional metadata fields, such as the retweet count.

5.4 Classification method

To train classifiers, we use the Maximum Entropy and the Linear Support Vector Machines algorithms[1], which in our previous study on the same dataset proved to be the top performing algorithms[17]. Based on the same study, we use classification features, which were found to positively contribute to the precision of the MaxEnt and SVM classifiers, in order to maximize the quality of automatically added train instances. The features included:

`Unigrams`: whitespace-separated word tokens (nominal: *please, help, fire*).

`Bigrams`: token sequences with the length of two (nominal: *was_scary, we_complained*).

`PartOfSpeechTags`: separate features are created from part-of-speech (PoS) categories, as assigned by a PoS tagger (nominal: *NNS, JJ, VBD*).

`ContainsHashtags`: whether or not the tweet contains any hashtags (Boolean).

`RetweetCount`: the number of times the message has been retweeted (continuous).

`ContainsURL`: whether the tweet contains a URL (Boolean).

Prior to training and classification, all features are converted to continuous values.

A usual experimental setup for text classification involves randomly splitting labeled data into a train and a test set. However in our experiments, to better reflect intended real-world use cases, the train-test split was done in a way that the test data contained tweets only on those disasters that were not included into the train data, i.e., simulating the conditions when a disaster needs to be detected before any manually labelled data relating to it are available. Thus we create nine train-test splits, so that in each split data on 23 disasters were used for training and data on 3 remaining crises were used for testing. The performance of the classifiers was measured in terms of precision, recall and F-measure rates averaged across the nine train-test splits. In the following sections, we report them only for the positive class that has main practical interest.

6 Results and Discussions

As mentioned in Section 4, we create two feature subsets to be used in the co-training algorithms: View 1, consisting of lexical features, and View 2, consisting of grammatical features and features derived from metadata found in the messages. To verify both views achieve reasonable performance, we evaluated them on the labelled data. Tables 2 and 3 show the results for the three classification tasks.

The results indicate that for the first two tasks both views have a similar level of accuracy: the differences between them are no more than 4 points for either precision and recall, for both MaxEnt and SVM. The eyewitness detection task, however, seems much harder than the other two. With MaxEnt, View 1 achieves noticeably better precision, while View 2 is better in terms of recall. This finding is in agreement with our previously published experiments [17] that showed that lexical features contribute to higher precision, while non-lexical ones improve recall. For SVM, the picture is the opposite: View 1 has a higher recall but lower precision than View 2.

We then compared the results achieved by a general classifier, i.e. one that is training on the entire train set of labeled data, to the ensemble and the co-training classification methods. These results for the Relatedness task are shown on Figure 2 (MaxEnt) and Figure 3 (SVM).

[1] We use the implementation in the scikit-learn library: http://scikit-learn.org/stable/

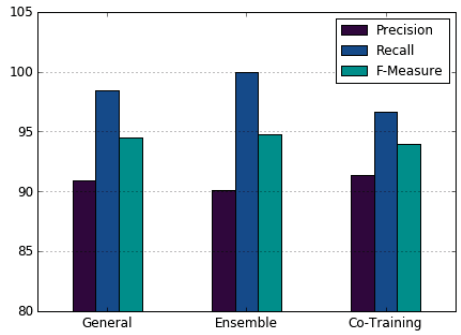


Figure 2. Performance of the general classifier, the ensemble method and the co-training method, Relatedness task, MaxEnt.

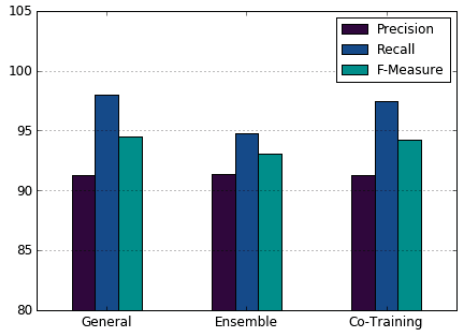


Figure 3. Performance of the general classifier, the ensemble method and the co-training method, Relatedness task, SVM.

The three methods perform very similarly, all achieving both precision and recall of over 90 points. The ensemble classifier improves on the general method by 4 points in terms of recall, but drops several points in precision, which results in an insignificant for F-measure. The co-training method fails to obtain any improvement on the general classifier, for both learning methods, trading a minor gain in precision for a somewhat greater loss in recall.

On the Informativeness task (Figures 4 and 5), the picture is similar. Co-training results for both precision and recall are almost the same as for the general method, the difference being no more than 1 point. The ensemble method, as in the Relatedness task, shows a five points gain on the general method in terms of recall, but loses nine points in terms of precision.

On the Eyewitnesses task (Figure 6 and 7), both ensemble and co-training improve on the general method in terms of F-measure, by 11 and 17 points, respectively. The ensemble method shows a particularly large increase in recall (by 62 points), although it also loses a lot in precision (53 points). The co-training method gains 14 points in recall, but loses 27 in precision.

Thus, the experimental results demonstrate that the Relatedness and Informativeness tasks are not affected by the data heterogeneity problem; a general classifier that can separate positive and negative classes with a high accuracy levels (F-measure of over 85 points) can be trained on modest amounts of data; ensemble or co-training methods offer no significant improvement over the baseline method. Although direct comparison with previous work is problematic, because previous studies used different evaluation datasets, evaluation metrics and slightly different definitions of "informativeness", the results we have attained on these two tasks are similar to previous studies who also aimed to detect informative tweets. Verma et al. [21] classified tweets into those that contribute and situational awareness and those that do not, finding that the best classification method achieves the accuracy of 0.88. Imran et al (2014) classify tweets into informative and non-informative, reporting the AUC rate of 0.8.

	View 1			View 2		
	Precision	Recall	F-measure	Precision	Recall	F-measure
Relatedness	90.1	99.9	94.6	90.87	97.8	94.1
Informativeness	81.8	93.2	87.1	83.4	89.7	86.3
Eyewitnesses	52.7	1.3	2.6	45.9	3.5	6.1

Table 2. Performance of MaxEnt classifiers trained on two views used in the co-training algorithm, on three classification tasks.

	View 1			View 2		
	Precision	Recall	F-measure	Precision	Recall	F-measure
Relatedness	90.4	99.0	94.4	90.0	99.8	94.6
Informativeness	84.6	89.0	86.7	83.9	89.3	86.4
Eyewitnesses	55.9	5.3	9.4	58.8	2.3	4.4

Table 3. Performance of SVM classifiers trained on two views used in the co-training algorithm, on three classification tasks.

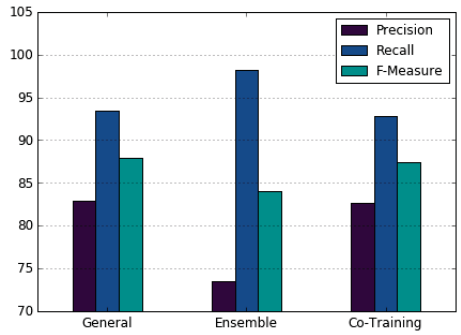


Figure 4. Performance of the general classifier, the ensemble method and the co-training method, Informativeness task, MaxEnt.

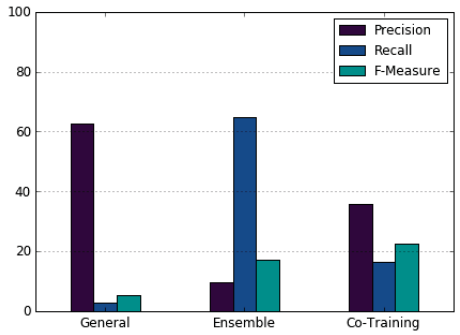


Figure 6. Performance of the general classifier, the ensemble method and the co-training method, Eyewitnesses task, MaxEnt.

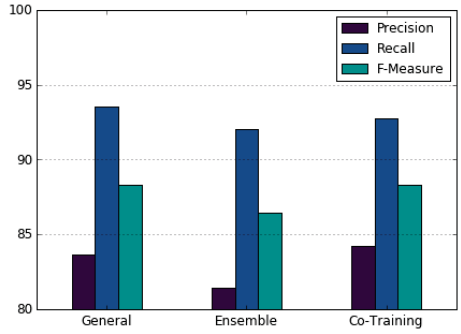


Figure 5. Performance of the general classifier, the ensemble method and the co-training method, Informativeness task, SVM.

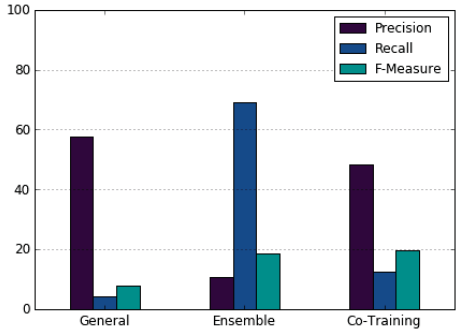


Figure 7. Performance of the general classifier, the ensemble method and the co-training method, Eyewitnesses task, SVM.

The Eyewitnesses task, however, proved a much harder problem. The baseline classifier achieves a precision rate of 60, but also an extremely low recall (under 10 points), which suggests that the model overfits and is not able to generalize sufficiently to the test data. The ensemble method makes it possible to improve recall to over 60 points, at the expense of precision, but nonetheless improving on the baseline in the F-measure. The co-training technique produces a similar effect, also beating the baseline in terms of the F-measure, which is also somewhat higher than that of the ensemble method. To our knowledge, Imran et al.'s study [8] is the only previous paper that evaluated the ability of a classifier to detect eyewitness accounts of mass emergencies. Their Naïve Bayes classifier achieved the F-measure of 60 points, also suggesting that this task is harder than that of informativeness or relatedness classification. These results are also higher than ours, but it should be noted that a direct comparison is not possible because of different experimental settings: specifically, Imran et al. [8] trained and tested their classifier on data representing the same mass emergency event.

7 Conclusion

In this paper we examined the task of detecting social media messages related to a mass emergency event, when the type of the event is not known at the training stage. We studied two ways to overcome the problem of data heterogeneity that affects the classifier in this situation: an ensemble classifier which combines predictions of classifiers specific to known types of disasters and a co-training method, which aims to reduce data heterogeneity by adding more train instances acquired from unlabeled messages in a bootstrapping manner.

In our experiments we studied three problems: detection of messages related to a disaster, detection of informative messages that contribute to situation awareness of first responders, and detection of first-hand accounts of the mass emergency events. We find that the first two tasks are relatively easy and good classification accuracy (an F-measure close to 80 or higher) can be achieved by a single generic classifier, even if the type of the disaster of the test data is not known in advance; ensemble or co-training methods did not offer any great advantage over the general classifier. The task of detecting eyewitness account proves much harder, but this is where the strengths of the ensemble and the co-training methods come to light: in comparison to using a general classifier, they both lead to significant gains in recall and F-measure.

Our results also suggest that there is considerable room for improvement for the both methods on the Eyewitness task. Thus, our future work will focus on exploring parameters of the ensemble and the co-training method, such as the effect of

the amount of automatically added training data, as well as other semi-supervised techniques such as active learning, in the context of this classification scenario. The proposed classification methods will eventually be incorporated into a practical system for detection and monitoring of mass emergency events on social media and evaluate their utility within simulated real-world use cases, where their benefits will be measured in terms of the efficiency of semi-automated detection of emergency situations by first responders, ultimately enabling early warning and more expedient recovery operations.

8 References

[1] Zahra Ashktorab, Christopher Brown, Manojit Nandi, and Aron Culotta. 2014. Tweedr: Mining Twitter to inform disaster response. In Proc. of ISCRAM.

[2] Hila Becker, Mor Naaman, and Luis Gravano. 2011. Beyond trending topics: Real-world event identification on Twitter. Proc. of ICWSM. 438–441.

[3] Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In Proceedings of COLT, pp.92–100.

[4] Cornelia Caragea, Nathan McNeese, Anuj Jaiswal, Greg Traylor, H. Kim, Prasenjit Mitra, Dinghao Wu, A. Tapia, Lee Giles, Bernard J. Jansen, and others. 2011. Classifying text messages for the Haiti earthquake. In Proc. of ISCRAM.

[5] Minmin Chen, Kilian Q. Weinberger, John C. Blitzer. 2011. Co-Training for Domain Adaptation. In Proc. NIPS-2011.

[6] Thomas G. Dietterich. 2001. Ensemble methods in machine learning. In Kittler, J., Roli, F., eds.: Multiple Classifier Systems. LNCS Vol. 1857, Springer (2001) 1–15.

[7] Yan-Shi Dong, Ke-Song Han. 2004. A comparison of several ensemble methods for text categorization. In Proc. IEEE International Conference on Services Computing. pp. 419-422.

[8] Muhammad Imran, Shady Elbassuoni, Carlos Castillo, Fernando Diaz, Patrick Meier. 2013. Extracting Information Nuggets from Disaster-Related Messages in Social Media. In Proc. of ISCRAM.

[9] Muhammad Imran, Carlos Castillo, Ji Lucas, Patrick Meier, and Sarah Vieweg. 2014. AIDR: Artificial intelligence for disaster response. In Proc. of WWW (Companion). IW3C2, 159–162.

[10] Rui Li, Kin Hou Lei, Ravi Khadiwala, and KC-C Chang. 2012. Tedas: A Twitter-based event detection and analysis system. In Proc. of ICDE. IEEE, 1273–1276.

[11] Benjamin Mandel, Aron Culotta, John Boulahanis, Danielle Stark, Bonnie Lewis, and Jeremy Rodrigue. 2012. A demographic analysis of online sentiment during hurricane Irene. In Proc. of the Second Workshop on Language in Social Media (LSM '12). Association for Computational Linguistics, Stroudsburg, PA, USA, 27-36.

[12] Huina Mao, Xin Shuai, Apu Kapadia. 2011. Loose Tweets: An Analysis of Privacy Leaks on Twitter. In Proc. WPES'11, Chicago, Illinois, USA.

[13] Adam Marcus, Michael S. Bernstein, Osama Badar, David R. Karger, Samuel Madden, and Robert C. Miller. 2011. Twitinfo: Aggregating and visualizing microblogs for event exploration. In Proc. of CHI. 227–236.

[14] Aibek Musaev, De Wang, and Calton Pu. 2014. LITMUS: Landslide detection by integrating multiple sources. Proc. of ISCRAM.

[15] Vincent Ng and Claire Cardie. 2003. Bootstrapping Coreference Classifiers with Multiple Machine Learning Algorithms.

[16] Alexandra Olteanu, Sarah Vieweg, Carlos Castillo. 2015. What to Expect When the Unexpected Happens: Social Media Communications Across Crises. In Proceedings of the ACM 2015 Conference on Computer Supported Cooperative Work and Social Computing (CSCW '15). ACM.

[17] Viktor Pekar, Jane Binner, Hossein Najafi, Chris Hale. 2016. Selecting Classification Features for Detection of Mass Emergencies on Social Media. In Proc. of the International Conference on Security and Management. Las Vegas, NV.

[18] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. 2010. Earthquake shakes Twitter users: Real-time event detection by social sensors. In Proc. of WWW. ACM, 851–860.

[19] Tom De Smedt and Walter Daelemans. (2012). Pattern for Python. Journal of Machine Learning Research. 13, 2063-2067.

[20] Mark Steedman, Miles Osborne, Anoop Sarkar, Stephen Clark, Rebecca Hwa, Julia Hockenmaier, P. Ruhlen, S. Baker, and J. Crim. 2003. Bootstrapping statistical parsers from small datasets. In Proc. the EACL.

[21] Sudha Verma, Sarah Vieweg, William J. Corvey, Leysia Palen, James H. Martin, Martha Palmer, Aaron Schram, and Kenneth Mark Anderson. 2011. Natural language processing to the rescue? Extracting "Situational Awareness" tweets during mass emergency. In Proc. of ICWSM.

[22] Jie Yin, Andrew Lampert, Mark Cameron, Bella Robinson, and Robert Power. 2012. Using social media to enhance emergency situation awareness. IEEE Intelligent Systems 27, 6, 52–59.

[23] Andrea Zielinski and Ulrich Buegel. 2012. Multilingual Analysis of Twitter News in Support of Mass Emergency Events. In of the 9th International ISCRAM Conference – Vancouver, Canada, April 2012.

An Application of Data Mining in Energy Industry

Jongsawas Chongwatpol
NIDA Business School, National Institute of Development Administration
jongsawas.c@ics.nida.ac.th

Extended Abstract/Poster Papers

1. INTRODUCTION

Many power producers are facing challenges from competitive pressure not only to ensure an uninterrupted and reliable energy supply but also to comply with the environmental regulation. The use of coal for electric power energy has addressed concerns about the environmental impact of the emissions of toxic substances [1-4], resulting in the environmental impact on climate and air quality. Although many technologies and advanced energy generation systems have been introduced and implemented to help controlling the level of toxic substance emissions to the environment, it is still very difficult to understand the energy consumption behavior and how the power generation process have a great impact on the quality of toxic emissions. This study seeks to fill this gap and outline a way to incorporate data mining techniques into existing coal-fired power plant data to monitor plant operations so that the level of toxic substance emissions complies with the mandatory standards for environmental protection.

2. COAL-FIRED POWER PLANT

A case study from a coal-fired power plant in Thailand has been conducted to explore the implication of data mining techniques to promote both corrective action and preventive maintenance at the power plant site. Currently, the plant is observed a great variation of the level of toxic substances and the top priority is to comply with the air pollutant emission standards. The plant is operated to provide electricity to the government with the maximum generating capacity of 1,434 megawatts. The process of the electricity generation starts when the crushed, powder-like, coal is burned at the very high temperature in the combustion chamber of a boiler. The hot gases produced then convert the water in the boiler into steam in order to spin the turbines to generate electricity. On the other hand, the byproduct of this combustion process is the flue gas, which contains many toxic substances such CO_2, NO_x, SO_x, or fly ash. This flue gas must be treated appropriately before being discharged into the air to comply with environmental standard. The power plant has installed, calibrated, and maintain a continuous toxic-substance monitoring system in the two stack outlets. However, the plant observed a great variation in NO_x readings from the flue gas exhaust emissions. Surprisingly, the NO_x readings on both outlets are dispersed differently even though the combustion process remains the same (see Fig 1 and Fig 2). Thus, the research question of this study is set up as follow: What are the important factors that have a great impact on the level of NO_x in the flue gas exhaust emissions on both stack outlets?

3. RESEARCH METHODOLOGY

To answer this research question, this study examine whether more complex analytical data mining techniques such as decision tree, neural network, stepwise polynomial regression, and support vector machine can better explain and predict the variation of NO_x level, which may be caused by differences in the operational control process or suppliers of coal. The dataset is partitioned into 50% for training and validation before applying those data mining techniques to predict and explain the leading causes of variation in NO_x level. The dataset contains variables from boilers, burner, turbine, pulverizer, pump, generator, transformer, and cooling water as well as chemical composition in coal such as Al_2O_3, CaO, Fe_2O_3, Ti_2O, and Mn_3O_4. We follow the CRISP-DM (Cross Industry Standard Process for Data Mining) model as a guideline for diagnosing the variation of NO_x level. CRISPDM breaks down this data mining project into six phases: business understanding, data understanding, data preparation, modeling, evaluation, and deployment [5].

4. DISCUSSION AND CONCLUSION

The results of this study not only provide a great signals of any unusual operational and coal-quality factors that influence the level of NO_x but also help explain and predict the leading causes of variation in the emission of NO_x in the combustion process. As presented in Fig.1 and Fig.2, a high NO_x level (above 241 ppm) requires immediate attention and corrective action is needed to comply with the environmental regulation. However, our interest is in the groups with an intermediate NO_x level, a level which is above an average of approximately 140 ppm but has not exceeded the threshold limit. This group can potentially shows signs of problems if the current electricity generation process is not changed or controlled. The stepwise regression and decision tree models show that main steam pressure, main steam temperature, Eco Outlet Gas O_2, Mn_3O_4, Ti_2O, and Na_2O are among factors that have a very high impact on the NO_x level. The plant manager can now minimize emissions of NO_x by promoting preventive maintenance on those key operational control process factors and coal properties, which consequently helps improve the overall performance of the power plant.

Keywords: Data Mining, Energy Industry, Coal-fired, NO_x

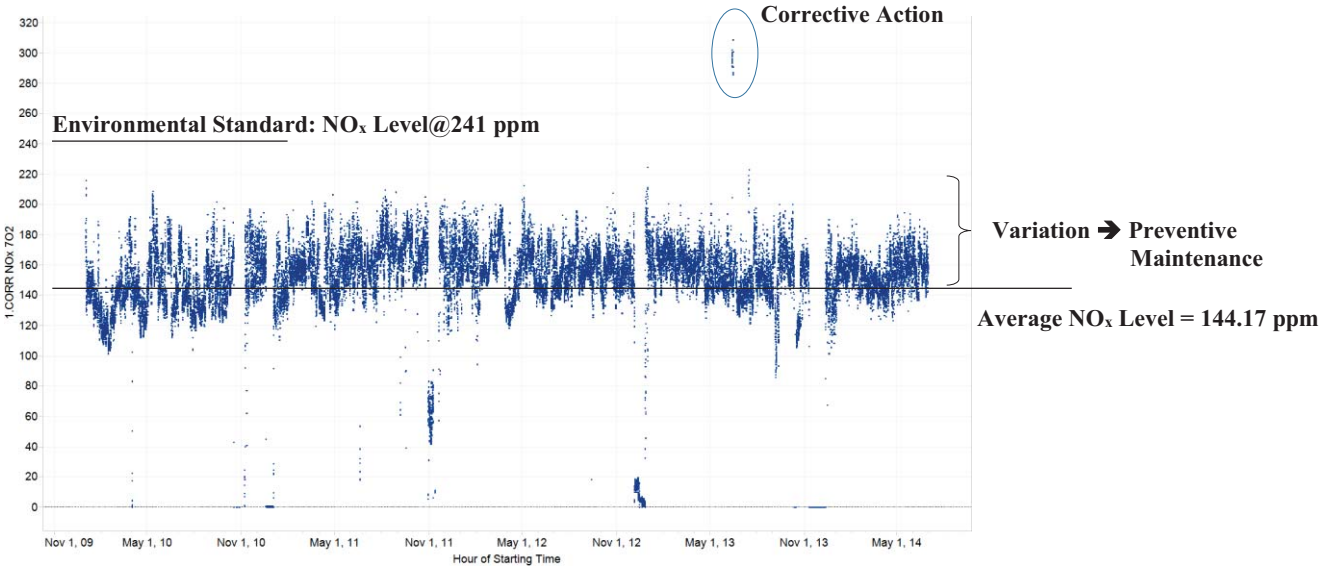


Fig. 1. NO_x level at the stack outlet #1

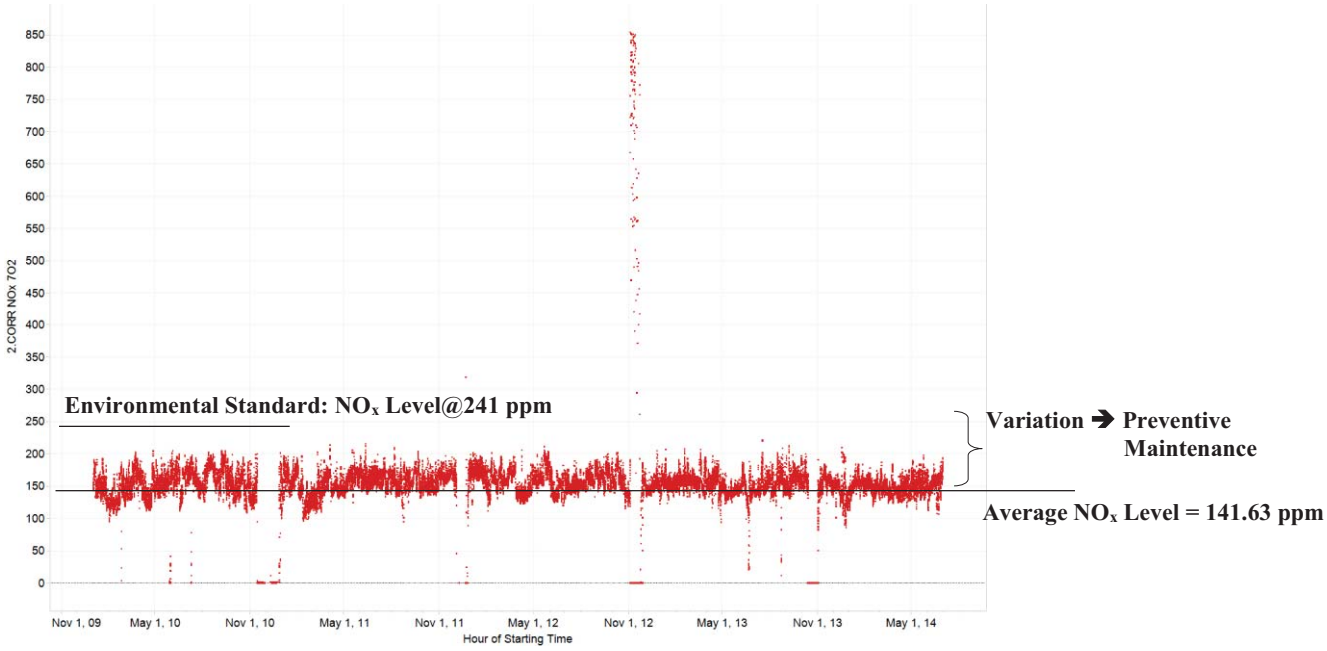


Fig. 2. NO_x level at the stack outlet #2

REFERENCES

[1] A. I. Chatzimouratidis and P. A. Pilavachi, "Decision support systems for power plants' impact on the living standard," *Energy Conversion and Management,* vol. 64, pp. 182-198, 2012.

[2] J. Chongwatpol and T. Phurithititanapong, "Applying analytics in the energy industry: A case study of heat rate and opacity prediction in a coal-fired power plant," *Energy,* vol. 75, pp. 463-473, 2014.

[3] A. Venkatesh, P. Jaramillo, W. M. Griffin, and H. S. Matthews, "Implications of changing natural gas prices in the United States electricity sector for SO_2, NO_X and life cycle GHG emissions," *Environmental Research Letters,* vol. 7, pp. 1-9, 2012.

[4] M. Waldner, R. Halter, A. Sigg, B. Brosch, H. Gehrmann, and M. Keunecke, "Energy from Waste - Clean, efficient, renewable: Transitions in combustion efficiency and NO_x control," *Waste management,* vol. 33, pp. 317-326, 2013.

[5] C. Shearer, "The CRISP-DM model: The new blueprint for data mining," *Journal of data warehousing,* vol. 5, pp. 13-22, 2000.

Zonification of Heavy Traffic in Mexico City

Ruben F. Estrada-S.[1], **Alejandro Molina**[2], **Adriana Perez-Espinosa**[3].
Araceli L. Reyes-C.[1], **Jose L. Quiroz-F.**[3], **and Emilio Bravo-G.**[1]
[1]Science and Technology College, Autonomous University of Mexico City, Mexico City
[2]The National Commission for Knowledge and Use of Biodiversity, Mexico City
[3]Electrical Engineering Department, Metropolitan Autonomous University, Mexico City

Abstract— *Mexico City has the most elevated traffic congestion in the world according to a recent report carried out by TomTom navigation products. A driver in Mexico City could spend up to 59% extra time trapped in traffic during a normal day. This paper applies a cluster analysis to find out the heavy traffic zones using massive data from the Social Navigation Network Waze. This zonification is fundamental for urban planning and transportation management, and may help develop accurate prediction models for heavy traffic.*

Keywords: Geo-spatial Data Analysis, Clustering, Vehicular Traffic, Data Mining

1. Introduction

The rapid urban growth of Mexico City has made it one of the most crowded in the world with a population close to twenty million people. As a result, the Metropolitan area is in constant expansion, forcing thousands of people to use their vehicles daily. Thus, this year the levels of air pollutants have exceeded the maximum exposure limits established by the World Health Organization (WHO) [1]. The heavy traffic in Mexico City causes a daily loss of 3.3 million man hours in traffic, which costs 33 billion pesos according to a study of The Mexican Institute for Competitiveness (IMC)[2].

An important characteristic of developing cities is the heavy traffic zones, where main roads converge and generate a large amount of traffic. The detection of such traffic zones allows seriously analyzing causes and proposals for realistic solutions. In the past, this problem has largely been approached by specialists with knowledge of the city. Nowadays, information technologies enable the detection of heavy traffic zones using massive and real time data. We refer to sensor, video and GPS devices [3].

The increasing use of smart devices with a GPS has promoted the development of different social navigation networks such as: Waze [4], Google Maps [5] and Inrix Traffic [6], etc. In Mexico City, the most popular of these networks is Waze, which collects data from user devices and allows them to report events and traffic levels, generating a real-time snapshot of the traffic situation.

This paper analyzes heavy traffic zones in Mexico City using data from Waze. In the light of Data Mining algorithms we propose a zonification of the metropolitan area based on massive quantities of user-reports. Section 2, shows an overview of the subject. In Section 3 the Waze network data is described. Results are presented in Section 4. Finally, the conclusion and future work are highlighted in Section 5.

2. Related Work

The problem of traffic congestion is a serious one in many cities the developing countries. In [7], V. Jain, A. Sharma and L. Subramanian presented an automated image processing mechanism for detecting the congestion levels in road traffic by processing CCTV camera image feeds. The algorithm is designed for noisy traffic feeds with poor image quality, based on live CCTV camera feeds from multiple traffic signals in Kenya and Brazil. Their analysis shows evidence of congestion collapse which lasts for long time-periods across multiple locations, allowing the detection of critical congestion zones. Furthermore, they gave causes of poor traffic management e.g. unplanned cities, poor discipline, alternate traffic means, archaic management and tighter budgets.

As for traffic studies in Mexico City in [8], the analysis of the traffic flow on the Mexico-City Cuernavaca highway is presented. They developed a traffic simulation system based on cellular automata with variable anticipation for single-lane traffic flow. Simulation results show that the congestion observed on days-off on this highway is not only a consequence of its complex topology. The main problem lies in the fact that there is an inefficient system of toll.

F.A. Armah,D.O. Yawson and A. Pappoe [9] applied system dynamics to identify related drivers, causes and effects of traffic problems in the Accra city, Ghana. Based on the analysis, they proposed policies, mainly economic instruments, to reduce traffic congestion in that city.

On the other hand, in [10] develops a new density-based algorithm to identify congested traffic routes, named FlowScan; instead of clustering the moving objects, road segments are clustered based on the density of the traffic they share in common.

3. Waze: A Social Navigation Network

Waze [4] is a noticeable social navigation network used by over 50 million users worldwide, which provide to its

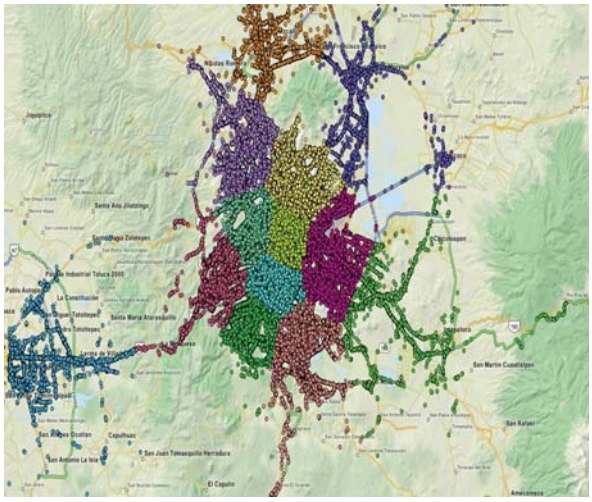


Fig. 1: A million datasets of heavy traffic reports were classified and projected using Lloyd's algorithm.

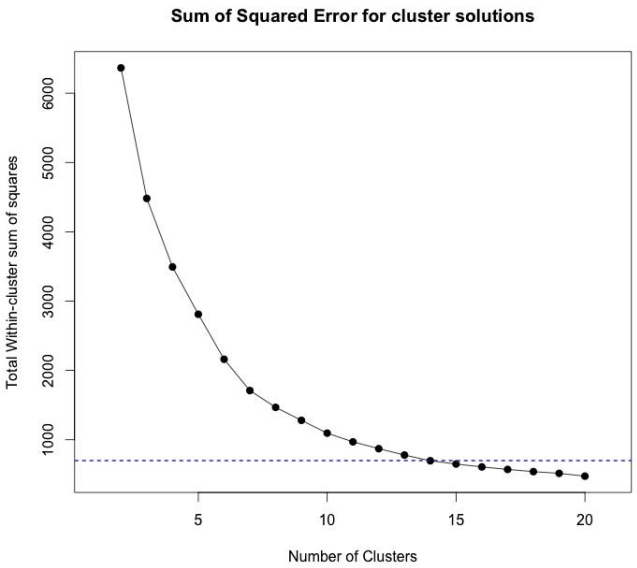


Fig. 2: Sum of squared error for different cluster solutions.

users real-time traffic information collected through a Waze application installed in smart devices with built-in GPS such as tablets or phones. Moreover, Waze application allows users to report traffic jams, accidents, speed and police traps. This information is processed to estimate the level of traffic on the roads, the time that a user will spend in traffic congestion and suggest alternative roads and the time estimated to arrive at a destination point. Waze has five level users: Baby, Grown-Up, Warrior, Knight and Royalty. As a user advances to high levels its reports have greater influence on the routing. In this work, we are interested in the traffic alerts, which can be of three type: moderate, heavy, and standstill.

4. Experiments and Results

In our study, we gathered the Waze public information using a web application which collects reports submitted by users. Our application focused on events in the Metropolitan Zone of Mexico City. The study area is bounded by the geographic coordinates North: 20.075, South=18.785, East=-98.582 and West=-99.659. More than a million reports were used to carry out the experiments. The reports have been collected every ten minutes since November 2015 but, as we want to model heavy traffic scenarios, we have selected the data when the number of traffic reports was greater than 12500 at the same time in the study area.

Figure 1 shows data points classified and projected, after a run of Lloyd's algorithm [11] to find a partition into well-shaped clusters. The initialization spans from $k = 2$ to $k = 20$ clusters after selecting several random sets of distinct points as the initial centers. Clustering results were used to define convex cells (hulls) that suggest a first approach for the zonification of the Metropolitan area of Mexico City in terms of real heavy traffic reports (see Figure 3).

However, two crucial questions immediately arise. First, what is the appropriate number of polygons? Second, how could we adapt our zonification to the existent political frontiers in the area? In the following sections these questions are discussed.

4.1 Clustering Evaluation

One common method of choosing the appropriate cluster solution is to compare the sum of squared error (SSE) for a number of cluster solutions. This is the sum of the squared distance between each member of a cluster and its cluster centroid. We have used SSE as a global measure of error. Figure 2 shows the SSE against a series of sequential cluster solutions. This useful graphical information allows us to choose an appropriate number of clusters. An appropriate cluster solution could be defined as the way at which the reduction in SSE slows dramatically. This produces an "elbow" in the plot of SSE against cluster solutions. In Figure 2, such "elbow" is found at the $k = 14$ cluster solution suggesting that $k > 14$ does not have a substantial impact on the total SSE.

4.2 Zonification of Vehicular Traffic in the Metropolitan area of Mexico City

Using official maps information from the National Institute of Statistics and Geography (INEGI[1]) we have assigned each municipality to its corresponding traffic zone (Figure 4). When more than one zone was assigned to the same

[1] *Áreas Geoestadísticas Municipales*, 2012, scale: 1:250000. INEGI. Online: http://www.conabio.gob.mx/informacion/metadata/gis/muni_2012gw.xml?&_xsl=/db/metadata/xsl/fgdc_html.xsl

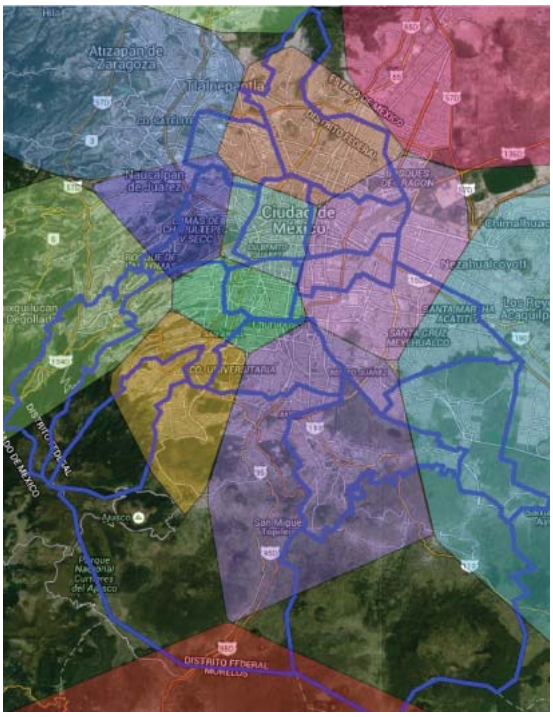


Fig. 3: The convex hull for each heavy traffic zone in the Metropolitan area of Mexico City.

municipality we have considered the zone density defined in equation 1.

$$\rho_k = \frac{\text{reports in k}}{\text{area of k } (km^2)} \quad (1)$$

Thus, given a municipality m and its assigned zones $Z_m : \{i \in 1 \dots k\}$, the final assigned zone is the one whose density is greater in the territory of m; i.e. $\arg\max_i \rho_i$. Note that the final proposed zonification is a product of official information and crowdsourcing data.

The definition of density (1) allows also to compare automatically the behavior of the same traffic zones in different situations. For example, we have assessed the impact of the public measures taken by the local government due to the poor air quality in recent days. Table 1 reports the density for each zone using all the data ρ_k as well as the density using only the data for the contingency period of time $\rho_{k'}$. The last column in Table 1 corresponds to the relative density between the normal traffic and the special period. We observe that the local government measures have had a great impact on traffic for some zones of the city, while for some others, the heavy traffic situation remains ($\rho_{kk'} \simeq 0$).

5. Conclusions and Future Work

Figure 4 presents the final proposed zonification map of vehicular traffic in the Metropolitan area of Mexico city. We have identified fourteen traffic zones in the analysed area.

k	$\rho_{k'}$	ρ_k	$\rho_{kk'}$
1	45.138	194.372	4.306
2	13.361	12.159	0.910
3	14.919	49.063	3.289
4	1.837	14.263	7.762
5	17.473	284.688	16.292
6	132.023	895.862	6.786
7	0.325	9.283	28.498
8	167.602	390.356	2.329
9	12.336	55.256	4.479
10	1.778	1.406	1.194
11	0.480	1.896	3.946
12	1.715	4.885	2.848
13	106.632	39.344	0.369
14	16.574	375.366	22.647

Table 1: Density of each heavy traffic zone in the Metropolitan area of Mexico City. Column 1 considers data only during the special contingency period and column 2 considers all the data.

Zonification is needed because clearly, the alerts of heavy traffic occur practically in all municipalities of the city. Moreover, thousands of heavy traffic reports are generated in almost any place in Mexico City at the same moment during normal days. As a consequence, a traffic model for all the city is too complex. For future work, we plan to use the proposed zonification to elaborate a finer analysis over each zone.

As part of a quality analysis we identify that Gustavo A. Madero, Miguel Hidalgo, Cuauhtemoc, Iztacalco and Benito Juarez municipalities are the most affected by traffic, because their access implies crossing many traffic zones. Data is consistent with the fact that the municipalities with less traffic are M. Contreras, Tlalpan and Milpa Alta, mainly in the south-western or south-eastern areas, since they are rural areas.

We conclude that the availability of traffic reports and official information could be processed with data mining algorithms in order to get insights to deal with such a complex problem as vehicular traffic. Our ambition is to go further and use data to create models of traffic prediction.

References

[1] Air-pollution-in-Mexico-City http://www.theguardian.com/world/2016/
[2] Viviendas para desarrollar Ciudades http://imco.org.mx/
[3] Honghui Dong, Mingchao Wu, Xiaoqing Ding, Lianyu Chu, Limin Jia, Yong Qin, Xuesong Zhou, Traffic zone division based on big data from mobile phone base stations, Transportation Research Part C: Emerging Technologies, Volume 58, Part B, September 2015, Pages 278-291, ISSN 0968-090X, http://dx.doi.org/10.1016/j.trc.2015.06.007.
[4] Waze https://www.waze.com
[5] Google Maps https://maps.google.com/
[6] Inrix http://www.inrix.com
[7] J. Vipin, A. Sharma, L. Subramanian. "Road traffic congestion in the developing world," ACM DEV 2012: 11:1-11:10.
[8] J.A. del Rio and M.E. Lárraga, . 'Transient Situations in Traffic Flow: Modelling the Mexico City Cuernavaca Highway," AIP Conference Proceedings;2005, Vol. 757 Issue 1, p190, April 2005.

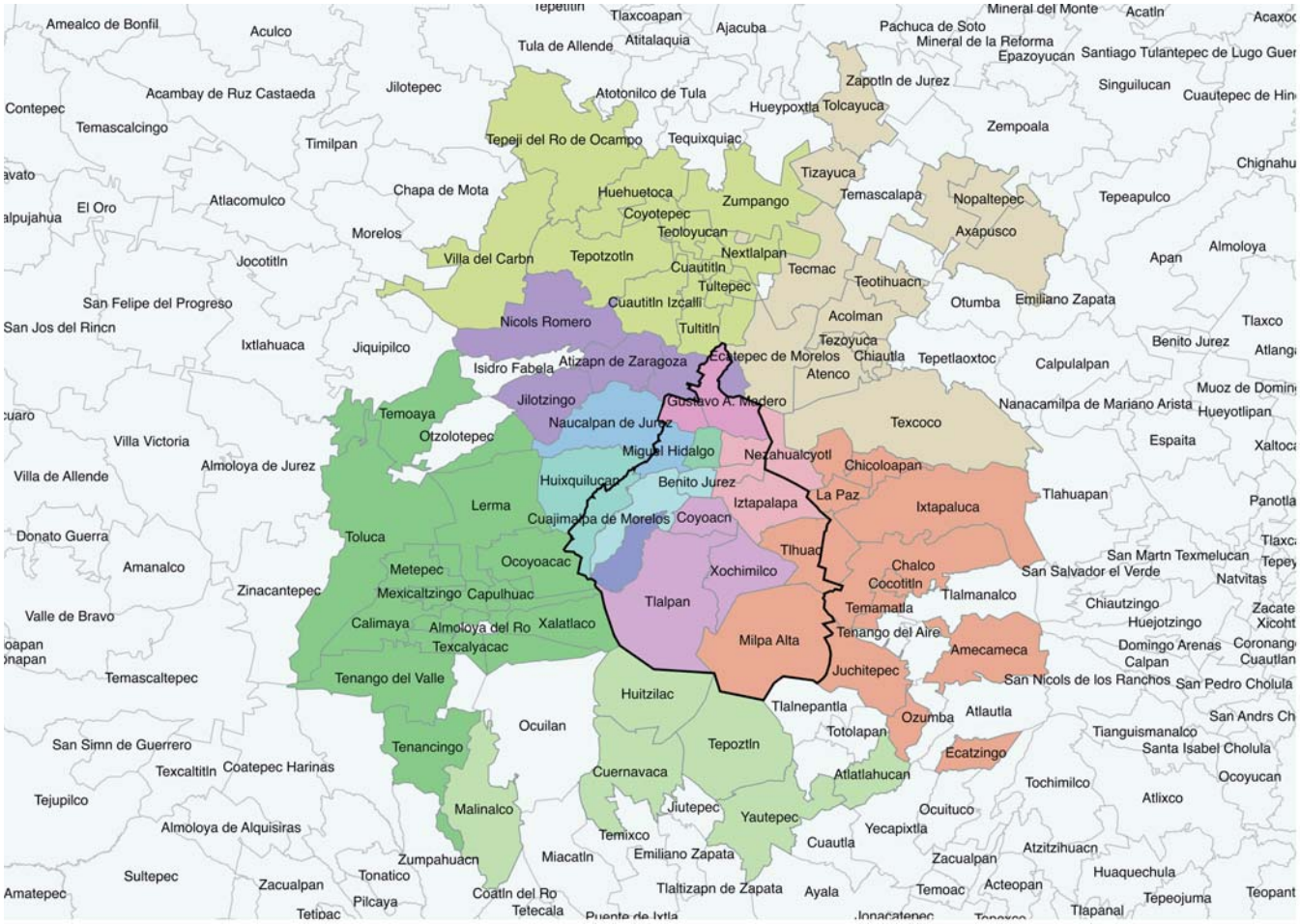


Fig. 4: Proposed zonification of Vehicular Traffic in the Metropolitan Area of Mexico city.

[9] Frederick A. Armah, David O. Yawson, and Alex A N M Pappoe. "A systems dynamics approach to explore traffic congestion and air pollution link in the city of Accra, Ghana." Sustainability, 2 (1):252–265, 2010. ISSN 20711050. doi: 10.3390/su2010252.

[10] X. Li, J. Han, J. G. Lee, and H. Gonzalez. 'Traffic density-based discovery of hot routes in road networks," In Proceedings of the 10th International conference on Advances in spatial and temporal databases, SSTD'07, pages 441–459, Berlin, Heidelberg, 2007. Springer-Verlag.

[11] Lloyd, S. P. Least squares quantization in PCM. Technical Note, Bell Laboratories. Published in 1982 in IEEE Transactions on Information Theory 28, 128–137.

Song Genre Classification via Lyric Text Mining

Anthony Canicatti
Computer and Information Science Dept., Fordham University, Bronx, NY, USA

Abstract— *Text mining is often associated with the processing of the natural English language to formulate a general conclusion about a body of text. In this work, this concept is applied in an attempt to build models that classify a song's genre based on its lyrics. Such models must be able to analyze the words that compose a song's lyrics and categorize the song as one of five genres: rock, rap, country, jazz or pop. This work features classifiers such as J48 decision trees, Random Forest, k-nearest-neighbor and Naive Bayes algorithms in order to classify each song. It also discusses the significance of data collection and pre processing in order to ensure valid results particularly when dealing with string values.*

Keywords: song, genre, classification, text, lyrics

1. Introduction

A song's lyrics generally hold information about what that song is about. To perform text mining on a song's lyrics is to use data mining techniques to learn what this general meaning is given the words in the song. This work attempts to take this a step further and build a classification model that uses the results from the text mining approach to categorize a song as either a rock, rap, country, jazz or pop song. This implies a certain number of assumptions were made about songs and their genres. Firstly, it assumes that songs within the same genre typically have similar lyrics. From a high level, it is possible, and sometimes trivial, to read the lyrics of two songs and conclude they belong to the same genre. Secondly, it assumes a song's genre is mutually exclusive. Herein lies the importance of the careful selection of genres to be classified. The five genres chosen are general enough that each genre's lyrics can be classified with high precision, but specific enough that each genre's lyrics are sufficiently distant from each other. In other words, the lyrics of rap songs are sufficiently distant from those of rock songs, but the lyrics of heavy metal songs are not sufficiently distant from those of rock. For the purposes of this work, it is assumed that these five genres do not overlap.

2. Data Collection and Preprocessing

Like in all data and text mining endeavors, collecting data and performing preprocessing techniques play a critical role in preparing the data for building models. In this work, a relatively large dataset containing sets of song lyrics along with their corresponding genres is necessary to begin training and testing a classification model.

2.1 Data Collection

The primary piece of data required in this work is the lyrical content of a number of songs from each of the five genres. Lists of songs from each genre were obtained using a script that queries an online music database. The music database[1] provides tables containing artist, title, time, BPM (beats per minute) and year information. The script creates a connection to the URL pertaining to the web page that contains the information above for each genre, extracts the table values from the HTML source, and generates a list of tuples of artists and song titles. This process is repeated for each genre, and in some cases, genres are amalgamated to account for the level of granularity of the genres. For example, the database has separate entries for Rap and R&B, two genres with differing musical qualities, but relatively equivalent lyrical content. Once a list of song-artist tuples is obtained for each genre, the list is fed to a Java application which uses an open source API to retrieve each song's lyrics from an openly accessible lyric website, MetroLyrics, and stores each lyric set in a CSV file. The lyrics found on this site are all input by users, creating a few possible issues in the dataset. The lyrics may not be correct, leading to issues in preprocessing. They also may not exist at all, in which case the Java application ignores the current input and continues on with the next tuple. The result of this process is a comma-separated data file with the attributes genre and lyrics, formatted as a single string of text, for each instance.

2.2 Preprocessing

Preprocessing is arguably the most critical component given the nature of this project. If data preprocessing is done correctly, we can expect that a classification model will produce consistent, meaningful results. Before importing the raw data file into Weka to begin applying advanced preprocessing techniques, some invalid patterns for each instance were removed manually in Excel. The lyric sets found on the MetroLyrics site often contained phrases indicating repitions. For example, if a section of lyrics was repeated twice, a lyric set may contain the string "x2" preceeding it. Furthermore, lyric sets often contained labeled sections, such as "[Chorus]", or "[Verse]". These phrases were removed using Excel's Replace feature, however, it is worth noting

[1] www.cs.ubc.ca/~davet/music/

that in instances where lyric repititions are denoted with a phrase like “x2", the lyrics themselves are not repeated, even though they would be in the song. This might cause for a skew in a classification model’s results.

2.2.1 Nominal to String

Weka provides a good deal of tools to clean raw data, particularly for use in text mining applications of traditional data mining learners. The first of which required in the dataset used in this work is the Nominal to String filter. The result of data collection is a raw CSV file of lyric and genre values. Weka parses all attributes in CSV files as nominal values, so the Nominal to String filter is used to transform the lyric attribute into a string (genre is already a nominal attribute, so it can be left as is).

2.2.2 String to Word Vector

The next step is vital: the String to Word Vector filter. This filter performs three critical operations all at the same time:

1) Split each string by the whitespace character so that it is transformed into a list of words
2) From the list generated above, eliminate any ‘stop’ words which would have no impact on a learner’s classification
3) Stem each word’s ending in order to preserve only the root

In the first step, each string of words is transformed into a list and each word becomes an attribute. For example, if a lyric set is the set “hello world”, the attributes ‘hello’ and ‘world’ are created. The second step is a common text mining technique where any stop words are eliminated completely from the dataset. Stop words include basic, meaningless words that have no impact on classification, for example ‘a’, ‘and’, ‘the’, etc. Weka provides a default list of stop words, which is used in this work. The third step is also a common text mining technique. In this operation, word endings are trimmed such that their tense is not a factor in determining the meaning of the word. For example, the words “jumping”, “jumped”, “jumps”, “jumper”, etc. all mean the same thing, jump, so all these words should be trimmed so that only “jump” remains, and they all become the same word. Weka provides several different stemming algorithms that perform this stemming. In this work, the IteratedLovins Stemmer is used. Previous to applying this filter, the data consists of two attributes and a list of instances. By splitting the strings into words, the filter turns the list of attributes into a unique list of the words contained in all instances, creating a master word list for the entire dataset. The instances are turned into counts of how many times each word, or attribute, appears in the instance’s song. Take the following as a simple illustration of how this works. Suppose there are two instances of lyric sets. Table 1 displays how the dataset would look before applying the String to Word vector filter.

Lyrics	Genre
The quick brown fox jumped over the lazy dog	Rock
I have a dog and a cat, the dog is a boy and the cat is a girl	Pop

Table 1: A sample dataset prior to running String to Word Vector

After running String to Word Vector, the words in these lyrics are extracted, stemmed and stop words are eliminated. Each instance becomes a count of how many times each word appears in the lyric set. Table 2 displays how the dataset would look after applying the filter.

quick	brown	fox	jump	over	laz	dog	two	cat	boy	girl	Genre
1	1	1	1	1	1	1	0	0	0	0	Rock
0	0	0	0	0	0	2	1	2	1	1	Pop

Table 2: The same dataset after running String to Word Vector

As shown in Table 2, the String to Word Vector filter converts the data to numerical data, making it easier for a classification learner to deal with. The three operations it performs on the raw data are the key factors in determining the consistency and meaningfullness in a model’s results.

2.2.3 SMOTE

The dataset collected using the procedure described in the previous section was originally very heavily class-imbalanced, due to the very nature of the means of collecting the data. The music database contains significantly more rock, pop and rap songs than it does country or jazz. After the initial pass of data collection, the largest class, rock, contained over 1700 entries, while the smallest class, jazz, contained about 200. Weka provides a means of handling such a problem from within the preprocessing stage, the SMOTE filter. This filter balances a given class value by producing instances in an intelligent way. SMOTE uses an adaptation of a k-nearest-neighbor algorithm which analyzes instances within a class, and creates new ones based on its closest neighbors. Ultimately, the dataset was balanced into sets of 500 instances for each class, meaning 500 song entries for each genre.

2.2.4 Principle Component Analysis

The final piece of preprocessing is by far the most mathematically intensive, as well as computationally heavy. In the example used above to illustrate the String to Word Vector filter, a dataset of 2 entries is used. The original

attribute set is of size 2, where the resulting attribute set after running the filter is of size 11, the number of unique, non-stop, stemmed words. Moreover, these two lyric sets are very small. Realistically, a single lyric set is about ten times the size of the ones used in the example. Therefore, the number of attributes when running the filter against the actual dataset is overwhelming. After obtaining a dataset of 2500 instances, the String to Word Vector filter created 8922 attributes. The Curse of Dimensionality is precisely this problem: as the number of dimensions, or attributes, grows, so too does the complexity of any model, and, typically, accuracy and meaningfullness of results declines. The remedy for such a problem is Principle Component Analysis (PCA). PCA involves dimensionality reduction by transforming the matrix of data using linear algebra techniques. First, the data is centered and its covariance matrix is obtained. The trace of the covariance matrix is known as the total variance (the diagonal entries of a covariance matrix are simply the variances of each attribute). A theorem of linear algebra states that the sum of the eigenvalues of a matrix equals its trace. Therefore, each eigenvalue of the covariance matrix corresponds to some percentage of the total variance. The eigenvalue with the highest percentage of the total variance is called the first principle component; the eigenvalue with the second highest percentage is the second principle component, and so on. PCA reduces dimensionality by eliminating lesser order principle components (small percentage eigenvalues) until a desired portion of the variance is maintained. Weka defaults this percentage to 95%. Therefore, eigenvalues of the covariance matrix will be removed until 5% of the total variance has been eliminated.

PCA transformed a dataset with 8922 attributes into a weighted dataset with 1339 attributes, a significant reduction, and, though not perfect, much more reasonable than the original attribute set.

3. Experiments

For all experiments, a percentage split was used to separate training data to build models on and testing data to evaluate them. 66% (1650 instances) of the data was used to train the model, and the remaining 34% (850 instances) to test it.

3.1 J48 Decision Trees

The first experiment performed on the preprocessed dataset was building a model using J48 decision trees, with pruning. Decision trees are popular methods of building classification models because of their ease of interpretation and clarity, but may not always be the best option depending on the type of data and attribute set. In this work, the dataset contains 1339 attributes, so a decision tree generated would be so large it would be impractical to try and scan it for suspicious nodes or meaningless splits, though its results may look plausible. Nonetheless, it is interesting to use this method regardless and examine how its results differ from those of other methods that try to account for these weaknesses.

3.2 Random Forest

The next experiment performed is building a Random Forest model. This method generates a number of decision trees at random and uses them in conjunction with each other when testing the model. Using this model typically generates a higher accuracy than a single decision tree because it allows for more specific splits, though may lead to a higher chance of overfitting data because of the higher number of splits. The initial run of the Random Forest algorithm was done generating 100 random trees, then 500 random trees. In the next section a comparison of the results from these two runs is given.

3.3 k-Nearest-Neighbor

The k-nearest-neighbor algorithm is sometimes called a lazy learner, as it's means of classifying an instance boils down to evaluating the k closest instances to it and picking a class based on those k instances, as opposed to building some rule set to follow when testing. There are various parameters that allow for customizing this kind of learner. First and foremost, k itself, the number of neighbors to consider when classifying an instance. However, another interesting parameter that can be experimented with is the means of evaluating distance. Two popular methods are classic Euclidean distance and Manhattan Distance. In these experiments, k was tested with values of 1 through 5, and both distance methods were tested.

3.4 Naive Bayes

The Naive Bayes algorithm is perhaps the most popular classification model used in text mining applications. It is fundamentally grounded in Bayes' Theorem, as follows:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{1}$$

where $P(A|B)$ is the probability of an event A, given the event B. This theorem provides an excellent relationship between prior and posterior probabilities, and therefore makes for an effective technique in classification. This theorem can be extended in the following way. Suppose a vector $x = (x_1, ..., x_n)$ represents n attributes, and some event C is to be predicted. According to Bayes' Theorem:

$$P(C|x) = \frac{P(x|C)P(C)}{P(x)} \tag{2}$$

This probability is most likely a more manageable value to compute. The Naive Bayes classification model uses this

approach to classify instances. The term 'Naive' reflects that the algorithm assumes all attributes are statistically independant. In the example above, this implies that all x_i in the vector $(x_1, ..., x_n)$ are uncorrelated with each other.

4. Results

For each experiment, the primary metrics evaluated are correctly classified instances, incorrectly classified instances, kappa statistic, mean absolute error as well as confusion matrices for each method. The kappa statistic is defined as:

$$\kappa = \frac{p_0 - p_e}{1 - p_e} \tag{3}$$

Also known as Cohen's kappa, this statistic compares observed accuracy with expected accuracy, where p_0 is observed accuracy and p_e is expected. A kappa value close to 1 indicates that observed accuracy is very high (some basic algebra shows that a kappa value of 1 denotes an observed accuracy of 100%). This can be a useful metric in determining the accuracy of the specific model built compared to the actual classification. Each entry in a confusion matrix is the number of times the class in the row is classified as the class in the column. In other words, the $(i, j)^{th}$ entry in the matrix is the number of times the i^{th} class was classified as the j^{th} class. Entries along the diagonal are the number of times a class was classified as itself. In a model with high accuracy, these numbers are expected to be the largest.

4.1 J48 Decision Trees

Table 3 displays general statistics for the results of building a J48 decision tree to classify the data, and Table 4 displays this model's confusion matrix.

Correctly Classified Instances	340
Incorrectly Classified Instances	509
Kappa Statistic	0.2509
Mean Absolute Error	0.2427

Table 3: General statistics for J48 Decision Tree

Country	Jazz	Pop	Rap	Rock	←Classified As
58	6	35	21	45	Country
26	**105**	13	7	22	Jazz
57	8	**52**	27	38	Pop
27	7	29	**76**	18	Rap
47	20	31	25	**49**	Rock

Table 4: J48 Decision Tree Confusion Matrix

This model was built in 57.22 seconds, contained 319 leaves, and was of total size 637. It generated an overall accuracy of 40.05%.

4.2 Random Forest

Table 5 displays general statistics for running Random Forest with generating 100 and 500 random decision trees.Table 6 displays this model's confusion matrix for running Random Forest with 100 and 500 random decision trees.

Correct	Incorrect	Kappa	Mean Abs. Err
100 trees			
368	481	0.2916	0.2565
500 trees			
402	447	0.3412	0.256

Table 5: Random Forest General Statistics

100 trees:					
Country	Jazz	Pop	Rap	Rock	←Classified As
73	0	30	16	46	Country
29	**101**	12	7	24	Jazz
56	0	**75**	21	30	Pop
37	0	38	**70**	12	Rap
63	0	44	16	**49**	Rock
500 trees:					
74	0	37	15	39	Country
23	**101**	14	5	30	Jazz
38	0	**87**	20	37	Pop
18	0	44	**78**	17	Rap
64	0	30	16	**62**	Rock

Table 6: Random Forest Confusion Matrices

Running the Random Forest with 500 random trees yields about a 4% increase in accuracy, as 100 trees generated an accuracy of 43.35% while 500 generated an accuracy of 47.35%.

4.3 k-Nearest-Neighbor

Figure 1 displays a plot of the accuracy of the k-Nearest-Neighbor algorithm with respect to k ranging from 1 to 5. As shown in the plot, the accuracy of the model is highest at k = 1 for both distance methods. As k increases, the accuracy tends to decline and stabilize around 30%. This phenomena is somewhat expected; it is likely that instances further away from the instance to be classified are not good predictors of the instance. In other words, it is more likely that one or two of the closest neighbors to an instance are better classifiers of that instance than its four or five closest neighbors.

Table 7 below displays accuracies as well as correctly classified instances for k from 1 to 5.

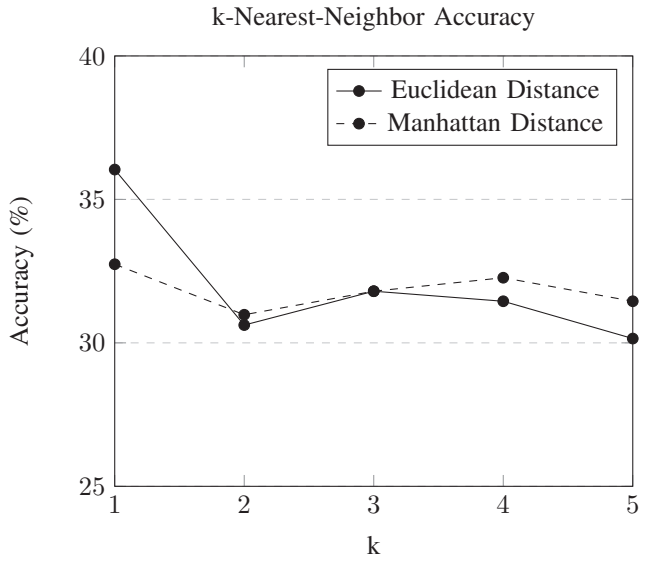


Fig. 1: Plot of kNN accuracy with respect to k

k	1	2	3	4	5
Accuracy (%)	36.04	30.62	31.8	31.45	31.15
Correctly Class.	306	260	270	267	256

Table 7: kNN accuracies and correctly classified instances

4.4 Naive Bayes

Table 8 displays general statistics for the results of a Naive Bayes classification model, and Table 9 displays this model's confusion matrix.

Correctly Classified Instances	263
Incorrectly Classified Instances	586
Kappa Statistic	0.1357
Mean Absolute Error	0.276

Table 8: General statistics for Naive Bayes

Country	Jazz	Pop	Rap	Rock	←Classified As
25	66	15	12	47	Country
21	**121**	6	3	22	Jazz
35	69	**29**	13	36	Pop
20	67	12	**32**	26	Rap
33	49	19	15	**56**	Rock

Table 9: Naive Bayes Decision Tree Confusion Matrix

The Naive Bayes classifier yielded an overall accuracy of 30.98%.

Table 10 below displays an overview of all models used. The parameters for each are those which yielded the highest accuracies. In the table, Random Forest refers to the model using 500 random decision trees and k-Nearest-Neighbor refers to the model with k equal to 1 using Euclidean distance.

J48	RandomForest	kNN	NaiveBayes
40.05%	47.35%	36.04%	30.98%

Table 10: Accuracies of all models

4.5 Results without Rap Genre

As discussed in the Data Collection and Pre Processing sections, one of the pressing issues pertaining to the dataset collected is the unreliability of the text data contained in each lyric set. Words are often misspelled, or unformatted, leading to non-distinctness in the String to Word Vector filter, as well as failure in the stemming algorithm. The rap genre is the foremost contributor to these instances of malformed or incorrect pre-processed data. To observe this genre's impact on the models' results, the genre was removed altogether and the previous experiments were performed on the data, now containing 2000 instances with 1157 attributes (after PCA). Table 11 displays accuracy results for J48 Decision Trees, Random Forest with 500 trees, k-Nearest-Neighbor with k=1 using Euclidean distance, and Naive Bayes.

J48	RandomForest	kNN	NaiveBayes
39.32%	51.69%	34.16%	34.46%

Table 11: Accuracy of Models without Rap Genre

As shown in the table above, Random Forest and Naive Bayes perform better with the rap genre removed, while J48 and kNN perform slightly worse. In the previous experiments, the per class error rates for the rap genre were 0.214 for kNN, 0.081 for Random Forest, 0.116 for J48 and 0.062 for Naive Bayes. Interestingly, Random Forest and Naive Bayes, the two models with the lowest error rate for the rap class, performed better after removing the genre, while the two with the higher rap error rates, J48 and kNN, performed slightly worse. However, it is worth noting that accuracy should expect to increase, regardless of any particular complexity within a specific class, whenever decreasing the number of class values. Considering a ZeroR approach, a simplistic model in which for each instance the most common class value is selected, the baseline accuracy is 20% for 5 class values and 25% for 4 class values, an increase in 5%. The increase in accuracy observed in the Random Forest model was just over 4%, so this increase nearly matches the expected increase when removing the class value.

5. Related Work

Music streaming services such as Pandora, Spotify, Google Play Music, etc. make use of data mining as a

means of providing subscribers personal playlists which are tailored specifically based on a user's preference in music. This is done by selecting songs a user listens to often, collecting metrics that quantify each song's musical qualities, using data mining to build a model that accurately captures the user's taste in music and providing the user with different songs that fit the model. Much work has been done using metrics that quantify things such as musical intensity, tempo, beat frequency, etc., but less have experimented with using lyrical text mining to perform a similar task. In one such work, the use of lyrics is provided *in addition* to the traditional methods to classify songs based on categories of moods (X. Hu, J. Downie, A. Ehmann). This work exhibits similar pre-processing techniques, including the removal of non-lyric text, but makes use of the lyrics in different ways. For example, it explains that function words (called stop-words in this work) actually exhibit predictive power in terms of text style analysis, and are used as an independent feature set, though yield worse results than other methods used. Furthermore, the work describes an approach called Bag-of-Words, in which lyrics are transformed into a set of unordered words, and features represent frequencies of each word, which is primarily how this work transforms lyrics into a feature set. Another approach is Part-of-Speech, in which words are grouped together based on their grammatical function in sentences. This approach provides more expressive power in that it may be helpful to analyze what type of words significantly impact classification, though may not necessarily yield better results. In another work, lyrics are used to find underlying emotional meanings in songs (D. Yang, W. Lee). This work uses 23 emotion categories such as power/strength/dominance vs. weak, active vs. passive, understatement vs. exaggeration, etc, in which songs are grouped into. Therefore, the approach used to generate a feature set is an adaptation of Part-of-Speech; rather than grouping words based on their grammatical function, they are grouped based on a pre-defined, arguably subjective, emotional function. Nonetheless, this work achieved an accuracy of 67% using an ensemble method in Weka.

6. Conclusions

The bulk of the work needed to begin performing any type of data mining experiments is contained in the pre-processing stage. As a result, the consistency and meaningfullness of the results obtained from the experiments is very much so contingent on the success of pre-processing. The results displayed in the previous section look poor at first glance, and it is without question they could be improved.

6.1 Future Work

The most glaring pre-processing error stems from the inconsistency of the lyrics contained in the rap genre (there is also a good deal of inconsistency in the rock and pop genres, but more so in the rap genre). One example of inconsistency is an instance where a word is spelled differently in two places, though each spelling neet not be 'correct'. The String to Word Vector filter only requires words to be spelled the same way, be it the correct spelling or not. In fact, as is true with text mining in general, the meaning and spelling of words is irrelevant - a model does not require a human understanding of the words in order to classify each instance. Nevertheless, it is obvious that these problems create issues for each model, and, as expected, accuracy jumps when removing the rap genre altogether. In future work, this kind of approach is optimal in attempting to achieve higher accuracies. In all experiments performed, Random Forest with 500 trees generated the highest accuracy at just over 50%. In order for all accuracies to improve, a more careful tweaking of the dataset needs to be performed. One such tweaking would be to remove all words not contained in the English dictionary (assuming the lyrics are all written in English). This approach is a bit extreme, and may not be necessary as the models themselves do not consider word meanings, but it would ease the pre-processing stage and create less inconsistency.

Another approach would be to re-evaluate the method of data collection altogether. The data is taken from an openly accessible lyric site, MetroLyrics. Given the high number of lyric sets required, it would be nearly impossible to verify the validity of each set of lyrics taken from the site and clean them to remove unwanted characters, misspellings, etc. Lastly, Principle Component Analysis clearly plays a critical role in the dimensionality reduction of text data, so it is also likely that a tweaking of this process may lead to better results. Nevertheless, it is clear that as the pre-processing of the dataset becomes more refined, the accuracies of each classification model improves.

References

[1] Xiao Hu, J. Stephen Downie, and Andreas F. Ehmann *Lyric Text Mining in Music Mood Classification*. *American music*, 2009.

[2] Dan Yang and Won-Sook Lee *Music Emotion Identification from Lyrics*. Multimedia, 2009. ISM '09. 11th IEEE International Symposium on, San Diego, CA, 2009

Using Machine Learning Algorithms to Improve the Prediction Accuracy in Disease Identification: An Empirical Example

Yong Cai, Dong Dai and Shaowen Hua

Abstract— **In the field of medicine, many diseases are difficult to diagnose. Patients may be free of symptoms for a long time before being discovered illness. This delay can lead to either life threatening incidence or high cost treatment such as liver transplant. Fortunately, technology development enables us to capture and store larger amount of diagnosis and treatment information. Also, various new machine learning algorithms have been developed and shown excellent performance in many circumstances. We use an empirical example to demonstrate how machine learning algorithms can help to improve prediction accuracy in identifying primary biliary cholangitis (PBC) patients. The models have been developed on real world patient diagnosis and treatment history data to predict PBC indication. We find in the example that Random Forest has outperformed AdaBoost or generalized linear models such as Logistic Regression and LASSO. By cross-validation, Random Forest has 0.942 compared to benchmark Logistic Regression 0.759 in prediction accuracy. This example shows about 24% of accuracy improvement over traditional method by selecting an appropriate algorithm. The result provides extra data point to cross-validate the claim that Random Forest is one of the general purposed algorithms that shall be considered in the analysis toolkit.**

Index Terms— **machine learning, Random Forest, Adaboost, algorithm comparison**

I. Introduction

Machine learning (ML) has been gaining lots of popularity in the recent decade. It's evolved from the field of artificial intelligence such as pattern recognition and computational learning. The recent popularity in ML is largely due to fast gaining computational power and ability of cost-effectively collecting and storing large amount of data. ML has been applied in many fields such as fraud detection, image and voice recognition etc. In this paper, we explore using machine learning algorithms to improve the prediction accuracy of disease identification. In an empirical example, we build predictive models to find primary biliary cholangitis (PBC) patients who haven't been diagnosed yet or miss PBC diagnose code.

PBC is a chronic, slow progressive disease that destroys the medium sized bile ducts in the liver [1], [2]. It is also known as primary biliary cirrhosis. PBC is primarily resulted from autoimmune destruction of the bile ducts. This causes bile acids (BA) to remain in the liver, where persistent toxics build up, and eventually damages the liver cell and causes cirrhosis. Disease progression in PBC is rather slow. Many patients with PBC may be completely free of symptoms, especially in the early stage of the disease. Because of this, the disease is often discovered through abnormal results on routine blood tests. In the United States, PBC is the one of the top ten causes of liver transplant [3], [4].

PBC is a rare disease. Boostra et. al. [5] did a systematic review by search 2286 abstracts. They concluded prevalence rates range from 19.1 to 402 per million. The prevalence and incidence rates vary greatly by geography. But the literatures they selected are based on small sample size ranging from 10 to 770 patients. Given the small observational sample, it is difficult to conduct epidemiology research at subnational level, such as finding the regional difference of prevalence and incidence. Predictive models can help identify more patients and derive a larger patient sample for further healthcare related research. In such cases, it is important to have a model provides accurate predictions to reduce the study bias.

Using patient age, gender, treatment pathway and dosage information in the data, we predict the likelihood that a patient has PBC disease. We applied Random Forest, Adaboost, LASSO, Naive Bayes algorithms as well as benchmark logistic regression model to test prediction accuracy.

The remaining of the paper will be arranged in following ways. In the next section, we describe the patient treatment and diagnosis database. In section 3, variable construction and modeling methodologies are presented. The following section applies and compares selected machine learning algorithms for predicting PBC patients. In the last section, the results and empirical findings are discussed.

II. Data Source Description

We extract the PBC training data from IMS medical claims database and longitudinal prescription (Rx) database. The claims data have diverse representation of geography, employers, payers, providers and therapy areas, with coverage of data from 90% of US hospitals, 80% of all US doctors. The data are also longitudinal, with 4 or more years of continuous enrollment.

The Rx data contains patient longitudinal prescription history. It is derived from electronic data received from pharmacies, payers, software providers and transactional clearinghouses. This information represents activities that take place during the prescription transaction and contains information regarding the product, provider, payer and geography. The Rx data is longitudinally linked back to an anonymous patient token and can be linked to events within the data set itself and across other patient data assets. Common attributes

Yong Cai is Director with IMS Health, Plymouth Meeting, PA 19462, USA (email: ycai@imscg.com).

Dong Dai is Senior Manger with IMS Health, Plymouth Meeting, PA 19462, USA (email: ddai@us.imshealth.com).

Shaowen Hua is Assistant Professor with School of Business, La Salle University, Philadelphia, PA, USA (email: hua@lasalle.edu).

and metrics within the Rx data include payer, payer types, product information, age, gender, 3-digit zip as well as the scripts relevant information including date of service, refill number, quantity dispensed and day supply. The Rx data covers up to 88% for the retail channel, 48% for traditional mail order, and 40% for specialty mail order.

All data are anonymous and HIPAA compliant to protect patient privacy.

III. Methodology

In the medical claims database, PBC can be identified by diagnosis ICD-9 code 577.6. However, medical claims database has limited longitudinal coverage or many PBC patients haven't been diagnosed yet. Our goal is to construct a training data including patients with complete diagnosis information and build a predictive model using patient past treatment history to identify PBC. Later on, we can use the model results to score and identify patients in the larger prescription database that doesn't contain ICD-9 flag.

Because Ursodiol is the only FDA approved drug to treat PBC, we restrict the study patients to Ursodiol prescribers only, in this way we capture the majority of the PBC patients who are not with 577.6 ICD-9 codes by only losing those who don't have either 577.6 ICD-9 codes or Ursodiol prescriptions.

A. Training Data

To build the training data, we select patients from medical claims database who have complete history of diagnosis and prescriptions. For those Ursodiol patients, we are able to identify whether they are PBC patients (with ICD-9 code 577.6) or not. The Ursodiol prescribers are selected if they have any prescription of Ursodiol during the one year selection window from August 2013 to July 2014. Index date is defined as their earliest date of prescribing Ursodiol in the selection window. The lookback period is one year back from the day before index date. The PBC indication (Y/N) is derived as whether the patient has ICD-9 code 577.6 during the lookback period. A total sample of 48,472 Ursidiol prescribers are included from claims database, among which there are 12,700 (26%) patients are with PBC disease.

B. Rx predictors for PBC

Since we are going to implement the predictive model trained from claims database to prescription (Rx) data, the predictors are derived only based on prescription activities, no diagnosis history will be included for prediction.

There are three types of predictors: 1) whether (Y/N) the patient has the treatment and 2) the number of fills of the prescriptions during the lookback period for a list of drug of interest, 3) other predictors include such as age, gender, Ursodiol's (on the index date) strength, dose (=strength x quantity/days supply), NDC (National Drug Code) and product name, specialty of physician who prescribed the Ursodiol on the index date. In total, we have 62 predictors.

TABLE I
Confusion Matrix

	Actual (Yes)	Actual (No)
Predicted (Yes)	True Positive (TP)	False Positive (FP)
Predicted (No)	False Negative (FN)	True Negative (TN)

TABLE II
Model Performance Metrics

Metric	Definition
AUC	Area under the ROC curve
Sensitivity	TP/(TP+FN)
Specificity	TN/(FP+TN)
Positive predictive value (PPV)	TP/(TP+FP)
Negative predictive value (NPV)	TN/(FN+TN)
Accuracy	(TP+TN)/(TP+FP+FN+TN)

C. Model Performance Evaluation

For binary classification problem, we have the confusion matrix (Table I) where "Yes" denotes positive class and "No" denotes negative class with any given machine learning algorithm.

With the confusion matrix, we use the following metrics (Table II) to evaluate model performance of predicting PBC with the derived Rx predictors, where receiver operating characteristic (ROC) curve is a graphical plot that illustrates the performance of a binary classifier system as its discrimination threshold is varied and ROC curve is created by plotting the true positive rate (TPR=TP/(TP+FN)) against the false positive rate (FPR=FP/(FP+TN)) at various threshold settings. And the area under the curve (AUC) is equal to the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one (assuming 'positive' ranks higher than 'negative').

D. Machine Learning Algorithms

For this binary classification (whether the patient has PBC disease or not) problem, Logistic Regression is applied as benchmark of model performance. The machine learning algorithms we consider in this paper include LASSO [6], Random Forest (also denoted by "RF" hereafter, [7]), Naive Bayes (also denoted by "NB" hereafter, [8]) and AdaBoost (also denoted by "AdaB" hereafter, [9]). We split the data into 80% (N=38,778) for training and 20% (N=9,694) for testing. Firstly, on the training dataset, we performed ten-fold cross-validation with all the machine learning algorithms to compare model performance and to choose a best algorithm; secondly, we train a predictive model by applying the best algorithm to the whole training dataset; lastly, we apply the predictive model to classify those patients prescribing Ursoldiol and without ICD-9 Code 577.6, where the cut-off probability is determined from training dataset to maximize the sum of Sensitivity and Specificity (graphically, with respective to the point closest to (0,1) on the ROC curve).

IV. Results and Conclusion

For all five algorithms, the averaged performance metrics (Table III, where cut-off probabilities are values that maximizing the sum of Specificity and Sensitivity on each training fold) and ROC curve (Figure 1) averaged (vertically) over ten-fold cross-validation on the training dataset is as following:

TABLE III
Model Performance (Ten-fold Cross-Validation)

	AUC	Sensitivity	Specificity
Logit	0.835	0.802	0.744
LASSO	0.836	0.793	0.752
NB	0.814	0.796	0.702
RF	0.983	0.939	0.943
AdaB	0.812	0.750	0.775

	PPV	NPV	Accuracy
Logit	0.527	0.914	0.759
LASSO	0.531	0.911	0.762
NB	0.487	0.906	0.727
RF	0.855	0.978	0.942
AdaB	0.542	0.897	0.768

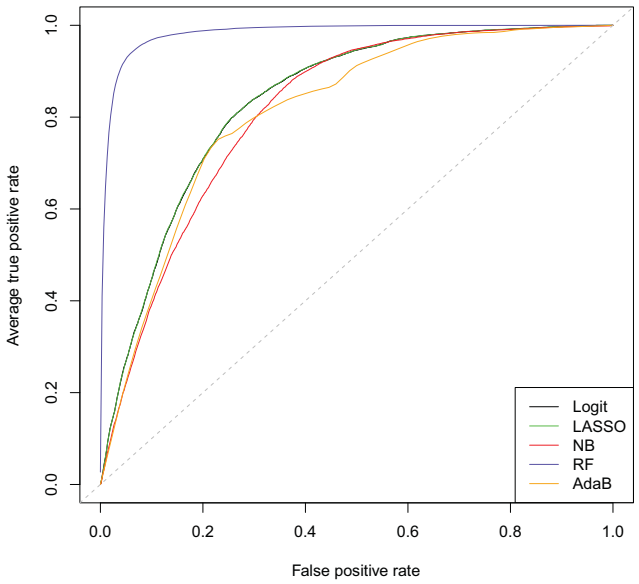


Fig. 1. Ten-fold Cross-Validation ROC curve

We can see that the Random Forest is the best one among all five algorithms, with an average AUC of 0.983 while that of Logistic Regression is 0.835, and other three algorithms of Naive Bayes, AdaBoost and LASSO are all with AUC less than 0.84. Note that on the above ROC curve figure, LASSO (green line) and Logistic Regression (black line) are almost overlapped, meaning that the variable selection process of LASSO mostly tends to select all the predictors resulting the similar performance as Logistic Regression, in other words, nearly each feature is contributing to the PBC disease prediction. And the significant performance improvement of Random Forest over Logistic Regression indicates the interactions among predictors, and in fact, for each treatment, the number of fills and flag (Y/N) are highly correlated, and the dose and strength are highly correlated to product name and NDC.

TABLE IV
Confusion Matrix and Performance Metrics of Classification on Testing Dataset

	Actual (Yes)	Actual (No)
Predicted (Yes)	True Positive (TP) $N = 2,321$	False Positive (FP) $N = 356$
Predicted (No)	False Negative (FN) $N = 219$	True Negative (TN) $N = 6,798$

Sensitivity	Specificity	PPV	NPV	Accuracy
0.914	0.950	0.867	0.969	0.941

Since Random Forest has the best performance among all five algorithms, we use it to train a predictive model with all the training dataset, then we use the cut-off probability that maximize the sum of Specificity and Sensitivity as cut-off value to classify whether a patient has PBC or not in the testing dataset. The confusion matrix ('Yes'=with PBC, 'No'=without PBC) and a list of performance metrics are as Table IV.

We also find that the performance of Random Forest on the testing dataset is consistent with the results of cross-validation on training dataset.

Our findings are also consistent with that of Caruana and Niculescu-Mizil (2006) [10] where Random Forest was one of the highest performers in their empirical example. As a general purpose ensemble method, Random Forest can pick complex variable dependencies and account for variable interactions. By averaging multiple decision trees and training on different parts of within the same training data, Random Forest was able to utilize the complex patient variables to a better extent and yield excellent accuracy (0.941) in cross-validation. Through the PBC empirical example, we provide one more data point to show that Random Forest is one of the most accurate general purposed learning techniques [11].

Acknowledgment

References

[1] R. Poupon, "Primary biliary cirrhosis: a 2010 update." *Journal of hepatology*, vol. 52, no. 5, pp. 745–758, 2010.
[2] G. M. Hirschfield and M. E. Gershwin, "The immunobiology and pathophysiology of primary biliary cirrhosis." *Annual review of pathology: Mechanisms of disease*, no. 8, pp. 303–330.
[3] K. D. Lindor, M. E. Gershwin, R. Poupon, M. Kaplan, N. V. Bergasa, and E. J. Heathcote, "Primary biliary cirrhosis." *Hepatology*, vol. 50, no. 1, pp. 291–308, 2009.
[4] C. Selmi, C. L. Bowlus, M. E. Gershwin, and R. L. Coppel, "Primary biliary cirrhosis." *The Lancet*, vol. 377, no. 9777, pp. 1600–1609, 2011.
[5] K. Boonstra, U. Beuers, and C. Y. Ponsioen, "Epidemiology of primary sclerosing cholangitis and primary biliary cirrhosis: a systematic review." *Journal of hepatology*, vol. 56, no. 5, pp. 1181–1188, 2012.
[6] R. Tibshirani, "Regression shrinkage and selection via the lasso." *Journal of the Royal Statistical Society., Series B (Methodological)*, pp. 267–288, 1996.

[7] L. Breiman, "Random forests." *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.
[8] T. Mitchell, "Generative and discriminative classifiers: naive bayes and logistic regression," 2005. [Online]. Available: https://www.cs.cmu.edu/ tom/mlbook/NBayesLogReg.pdf
[9] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting." *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.
[10] R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms." in *Proceedings of the 23rd international conference on Machine learning*, pp. 161–168.
[11] R. Genuer, J. M. Poggi, and C. Tuleau, "Random forests: some methodological insights." *arXiv preprint*, 2008.

Random Under-Sampling Ensemble Methods for Highly Imbalanced Rare Disease Classification

Dong Dai, and Shaowen Hua

Abstract— **Classification on imbalanced data presents lots of challenges to researchers. In healthcare settings, rare disease identification is one of the most difficult kinds of imbalanced classification. It is hard to correctly identify true positive rare disease patients out of much larger number of negative patients. The prediction using traditional models tends to bias towards much larger negative class. In order to gain better predictive accuracy, we select and test some modern imbalanced machine learning algorithms on an empirical rare disease dataset. The training data is constructed from the real world patient diagnosis and prescription data. In the end, we compare the performances from various algorithms. We find that the random under-sampling Random Forest algorithm has more than 40% improvement over traditional logistic model in this particular example. We also observe that not all bagging methods are out-performing than traditional methods. For example, the random under-sampling LASSO is inferior to benchmark in our reports. Researchers need to test and select appropriate methods accordingly in real world applications.**

Index Terms— **imbalanced, rare disease, random under-sampling, random forest**

I. Introduction

Rare diseases have low prevalence rates and they are difficult to diagnose and identify. The "Rare Disease Act 2002" defines rare disease to be less than 200,000 patients, or 1 in 1,500 [1]. By this nature, the rare disease dataset is extremely imbalanced. To predict or classify rare diseases from a large population is a challenging task due to low signal-to-noise ratio. This problem is called imbalanced learning in data mining field [2]. Classic statistical methods or standard machine learning algorithms are biased toward larger classes. Therefore, it is hard to positively identify rare disease patients who are in the minority class. If not treating and measuring properly, most of rare disease patients will be mis-classified as the other classes albeit the overall accuracy rate may appear to be high.

There are dedicated new algorithms and methods developed for imbalanced datasets classification. In this paper, we explore selecting and applying imbalanced machine learning algorithms to identify rare disease patients from real-world healthcare data. We will study the performance improvement from using imbalanced algorithms and compare the results to traditional methods. The empirical application we select is to identify Hereditary Angioedema (HAE) disease. HAE is a rare, genetic disease that causes episodic attacks of swelling under the skin. The prevalence of HAE is between 1 in every 10,000 and 1 in every 50,000 [3], [4].

Because it is rare, physicians have rarely encountered patients with this condition. On top of this, HAE attacks also resemble other forms of angioedema. Both make HAE hard to diagnose. Late or missed diagnosis can lead to incorrect treatment and unnecessary surgical intervention. Our objective in this application is to improve HAE patient identification rate using patient historical prescription and diagnosis information. Given the extremely unbalanced classes in the dataset, it is interesting to see how existing algorithms perform in such conditions. We will test a commonly used approach in imbalanced learning - Random Under Sampling method for ensemble learning with LASSO or Random Forest as element learners.

The remaining of the paper will be arranged in following ways. Section 2 describes the data source used in this empirical example. In section 3, we present the rules and methodologies in anonymous patient selection and variable construction. Next section introduces imbalanced learning algorithms. In the last two sections, the results are presented and discussed.

II. Data Source Description

To study the potential application of imbalance algorithms on rare disease identification, we select Hereditary Angioedema disease as an empirical use case. The data has been extracted from IMS longitudinal prescription (Rx) and diagnosis (Dx) medical claims data.

The Rx data is derived from electronic data collected from pharmacies, payers, software providers and transactional clearinghouses. This information represents activities that take place during the prescription transaction and contains information regarding the product, provider, payer and geography. The Rx data is longitudinally linked back to an anonymous patient token and can be linked to events within the data set itself and across other patient data assets. Common attributes and metrics within the Rx data include payer, payer types, product information, age, gender, 3-digit zip as well as the scripts relevant information including date of service, refill number, quantity dispensed and day supply. Additionally, prescription information can be linked to office based claims data to obtain patient diagnosis information. The Rx data covers up to 88% for the retail channel, 48% for traditional mail order, and 40% for specialty mail order.

The Dx data is electronic medical claims from office-based individual professionals, ambulatory, and general health care sites per year including patient level diagnosis and procedure

Dong Dai is Senior Manager with Advanced Analytics, IMS Health, Plymouth Meeting, PA 19462, USA (email: ddai@us.imshealth.com).

Shaowen Hua is Assistant Professor with School of Business, La Salle University, Philadelphia, PA, USA (email: hua@lasalle.edu).

information. The information represents nearly 65% of all electronically filed medical claims in the US. All data is anonymous at the patient level and HIPAA compliant to protect patient privacy.

III. Methodology

Since HAE disease is difficult to identify, many patients with this condition don't have a positive diagnosis ICD-9 code associate with them. Our objective was to build predictive models to find such patients using their past prescription records and other diagnosis histories. We extracted our training samples from Rx and Dx data sources described before.

A. Sample selection

HAE patients are selected as those with HAE diagnosis (ICD-9 CODE = 277.6) and at least one HAE treatment (prescription or procedure) during the selection period (1/1/2012 - 7/31/2015). The patient's index date is defined as the earliest date of HAE diagnosis or HAE treatment. Then the lookback period is defined as from earliest diagnosis or prescription date available from January 2010 till the day before index date. Using this rule, the selected HAE patients will have lookback periods with variable lengths. After further data cleaning by deleting the records without valid gender, age and region etc., we have 1233 HAE patients in the sample data.

Non-HAE patients are selected by randomly matching 200 non-HAE patients with similar lookback period for each one of the 1233 HAE patients. For example, if an HAE patient has a lookback period from 1/15/2013 to 2/5/2014, then a non-HAE patient is qualified to match and to be selected if he/she had clinical activity (any activity of prescription, procedure or diagnosis) from any day in January 2013 to any day in February 2014. Then the lookback period for the non-HAE patient will be defined as from earliest clinical activity date in January 2013 till latest clinical activity date in February 2014. This process of non-HAE patients matching to HAE patients has been done in a greedy manner. Specifically, non-HAE patient sample starts as an empty set, then 200 non-HAE patients matched to a given HAE patient are added to the non-HAE sample, and this approach is repeated for each HAE patient in the sample until we find 200 distinctive non-HAE patients for each HAE patient.

The above process generates 1233 HAE patients and 246600 (200 times 1233) non-HAE patients in the final patient sample.

B. Predictors for HAE

Literatures about HAE has been reviewed, from which a list of numerous potential HAE predictors are prepared. Specifically, three classes of clinical indications are derived for the lookback period for each patient, including prescriptions, procedures and diagnosis. Each clinical indication yields three predictors for the lookback period: flag (Yes/No: whether the patient had prescriptions, procedures or diseases), frequency (how many times the event of prescriptions, procedures or diseases occurred), average frequency (frequency divided by length of lookback period). These clinical predictors with demographic predictors (age, gender and region) compose the final predictors list.

TABLE I
Confusion Matrix

	Actual=1	Actual=0
Predicted=1	True Positive (TP)	False Positive (FP)
Predicted=0	False Negative (FN)	True Negative (TN)

TABLE II
Model Performance Metrics

Metric	Definition
AUC	Area under the ROC curve
AUPR	Area under the PR curve
Recall	TP/(TP+FN)
Precision	TP/(TP+FP)
True Positive Rate (TPR)	TP/(TP+FN)
False Positive Rate (FPR)	FP/(FP+TN)
Accuracy	(TP+TN)/(TP+FP+FN+TN)

C. Model Performance Evaluation

In a binary decision problem, a classifier labels data sample as either positive or negative. The decision made by the classifier can be represented in a structure known as a confusion matrix (Table I, "1" for positive class and "0" for negative class). The confusion matrix has four categories: True Positives (TP) are examples correctly labeled as positives; False Positives (FP) refer to negative examples incorrectly labeled as positive; True Negatives (TN) correspond to negatives correctly labeled as negative; finally, False Negatives (FN) refer to positive examples incorrectly labeled as negative.

Based on the confusion matrix, we will be able to further define several metrics to evaluate model performance as listed in Table II.

In ROC space, one plots the False Positive Rate (FPR) on the x-axis and the True Positive Rate (TPR) on the y-axis. The FPR measures the fraction of negative examples that are misclassified as positive. The TPR measures the fraction of positive examples that are correctly labeled. In PR space, one plots Recall on the x-axis and Precision on the y-axis. Recall is the same as TPR, whereas Precision measures that fraction of examples classified as positive that are truly positive.

D. Imbalanced Classification

Imbalanced data sets (IDS) correspond to domains where there are many more instances of some classes than others. Classification on IDS always causes problems because standard machine learning algorithms tend to be overwhelmed by the large classes and ignore the small ones. Most classifiers operate on data drawn from the same distribution as the

training data, and assume that maximizing accuracy is the principle goal [5], [6].

Therefore, many solutions have been proposed to deal with this problem, both for standard learning algorithms and for ensemble techniques. They can be categorized into three major groups [7], [8]: (i) Data sampling: In which the training data are modified in order to produce a more balanced data to allow classifiers to perform in a similar manner to standard classification [9], [10]; (ii) Algorithmic modification: This procedure is oriented towards the adaptation of base learning methods to be more attuned to class imbalance issues [11]; (iii) Cost-sensitive learning: This type of solutions incorporate approaches at the data level, at the algorithmic level, or at both levels combined, considering higher costs for the misclassification of examples of the positive class with respect to the negative class, and therefore, trying to minimize higher cost errors [12], [13].

In this paper we implement the Random-Under-Sampling (RUS) (majority) approach. We first randomly under-sample the majority class data and combine them with the minority class data to build an artificial balanced dataset, upon which machine learning algorithms will be applied. This process is repeated for several iterations, with each iteration generating a model. The final model is an aggregation of models over all iterations (See Algorithm 1, similar to bagging [14]). In this paper, we apply RUS with LASSO [15] and Random Forest [16] (hereby denoted as "Bagging LASSO" and "Bagging RF" respectively). Specifically, we firstly perform a random under sampling of the majority pool (non-HAE) and combine it with all HAE patients to build a artificial balanced dataset, then LASSO or Random Forest is applied to the balanced sampled data; models are aggregated over iterations of random samples to learn the predictive pattern of HAE, while accounting for possible interactions among predictors. The conventional logistic regression (denoted by "Logit" hereafter) is implemented as benchmark for model performance comparison.

The usual model performance metrics such as prediction accuracy and area under the ROC curve (AUC) are not appropriate for imbalanced classification problem. For example, the imbalance ratio is 1/200 (each one HAE patient has 200 matched non-HAE patients) in our data, a classifier which tries to maximize the accuracy of its classification rule may obtain an accuracy of 99.5% by simply ignoring the HAE patients, with the classification of all patients as non-HAE. Instead, we will use AUPR and Precision at various Recall levels for model performance comparisons in this paper. An important difference between ROC space and PR space is the visual representation of the curves. Looking at PR curves can expose differences between algorithms that are not apparent in ROC space [17].

For validation purpose, data is split into 80% for training and 20% for testing. Model performance for five-fold Cross-Validation on training data and further validation on testing data are both reported. The results and validation are presented in the next section.

IV. Results

With the 80% training data, we have $n = 986$ HAE patients and their 200 times matched non-HAE patients ($n = 197,200$). Then we perform five-fold cross-validation with the training data, specifically, we split all the training data into five folds, and for each given fold, we train a model with the remaining four folds and calculate the performance metrics on the given fold, the final performance outputs are metrics averaged over the five folds. Summary of the results are listed in Table III.

TABLE III
Model Performance (Five-fold Cross-Validation)

Metric	Logit	Bagging LASSO	Bagging RF
AUC	79.81%	83.03%	82.52%
AUPR	9.59%	9.21%	11.61%
Precision (Recall=50%)	4.49%	5.84%	5.99%
Precision (Recall=45%)	6.33%	7.23%	7.55%
Precision (Recall=40%)	7.86%	8.85%	10.04%
Precision (Recall=35%)	10.67%	10.10%	13.44%
Precision (Recall=30%)	13.61%	11.74%	16.98%
Precision (Recall=25%)	15.11%	13.81%	21.74%
Precision (Recall=20%)	18.46%	15.27%	24.69%
Precision (Recall=15%)	22.82%	18.95%	30.25%
Precision (Recall=10%)	29.02%	27.90%	31.83%
Precision (Recall=5%)	36.60%	34.02%	38.63%

We can see that Bagging RF has an improvement of 21% in terms of AUPR compared to Logistic Regression. And particularly for Recall level at 25%, Logistic Regression has a Precision of 15.11% while that of Bagging RF is 21.74%, which is more than 40% improvement over that of Logistic Regression.

And we can see PR curve (Figure 1) differentiates the algorithms better than ROC curve (Figure 2) (The dashed lines in two figures reflect the performance of a random classifier).

We further validate the model performances by applying the models trained from all training data to testing data, which has $n = 247$ HAE patients and their 200 times matched non-HAE patients ($n = 49,400$). A summary of the testing results are listed in Table IV. It further validates that Bagging RF outperforms Logistic Regression in this imbalanced classification problem.

The Bagging LASSO is inferior to Logistic Regression in our case. The reason could be that the high dimensional features have much correlation and in this case less sparse model is preferred.

In searching for better performance, we also test cost-sensitive learning methods such as weighted SVM and weighted LASSO on the same training data. Both of the results don't show improvement over the standard or random-under-sampling ensemble methods.

V. Conclusion and Discussion

In this rare disease identification problem, we test under-sampling and cost-sensitive learning algorithms on a real

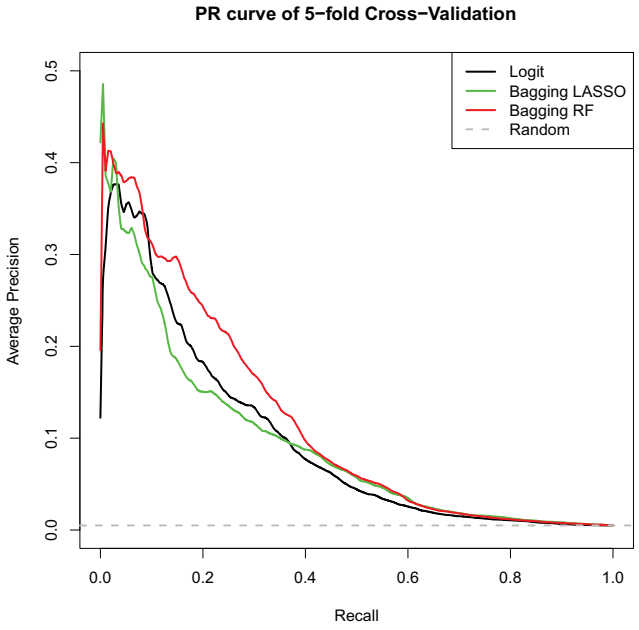


Fig. 1. Five-fold Cross-Validation PR curve

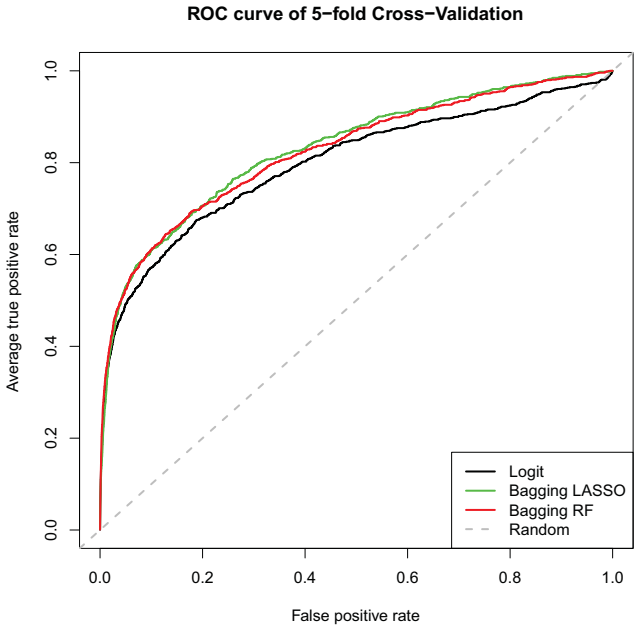


Fig. 2. Five-fold Cross-Validation ROC curve

TABLE IV
MODEL PERFORMANCE (TESTING)

Metric	Logit	Bagging LASSO	Bagging RF
AUC	82.77%	84.33%	85.12%
AUPR	13.02%	11.37%	14.09%
Precision (Recall=50%)	6.09%	5.63%	7.17%
Precision (Recall=45%)	8.35%	8.35%	8.43%
Precision (Recall=40%)	9.71%	9.02%	11.19%
Precision (Recall=35%)	12.63%	10.36%	14.48%
Precision (Recall=30%)	16.78%	12.85%	20.16%
Precision (Recall=25%)	21.91%	16.76%	22.55%
Precision (Recall=20%)	24.50%	20.00%	30.82%
Precision (Recall=15%)	24.03%	20.67%	37.76%
Precision (Recall=10%)	44.64%	37.31%	38.46%
Precision (Recall=5%)	54.55%	63.16%	57.14%

Algorithm 1 A general framework of Random-Under-Sampling (RUS) ensemble learning

Let S be the original training set.

for $k = 1, 2, \ldots, K$ **do**

Construct subset S_k containing instances from all classes with same number by executing the following: Random sample instances with (or without) replacement at the rate of N_c/N_i where N_c denote the desired sample size and N_i denotes the original sample size of class i.

Train a classifier $\hat{f}_k$ from subset S_k.

Output final classifier $\hat{g} = \text{sign}\left(\sum_{k=1}^{K} \hat{f}_k\right)$.

end for

world patient data case. The training sample contains only 0.5% positive patients and it is a good real world example to demonstrate imbalanced learning challenge. We find that in this empirical example, random-under-sampling random forest method can boost precision at various recall levels compared to standard models. In the reported table, it has 40% improvement in AUPR over the benchmark model.

In real world application, due to the data high dimensionality and case complexity, it is not guaranteed that random-under-sampling Random Forest always to be the recommended method. Researchers should try various methods and select most appropriate algorithm based on performance.

APPENDIX

ACKNOWLEDGMENT

REFERENCES

[1] "Rare diseases act of 2002 public law 107280 107th congress."

[2] H. He and Y. Ma, Eds., *Imbalanced learning: foundations, algorithms, and applications.* John Wiley and Sons., 2013.

[3] M. Cicardi and A. Agostoni, "Hereditary angioedema." *New England Journal of Medicine*, vol. 334, no. 25, pp. 1666–1667, 1996.

[4] L. C. Zingale, L. Beltrami, A. Zanichelli, L. Maggioni, E. Pappalardo, B. Cicardi, and M. Cicardi, "Angioedema without urticaria: a large clinical survey." *Canadian Medical Association Journal*, vol. 175, no. 9, pp. 1065–1070, 2006.

[5] N. V. Chawla, N. Japkowicz, and A. Kotcz, "Editorial: special issue on learning from imbalanced data sets." *ACM Sigkdd Explorations Newsletter*, vol. 6, no. 1, pp. 1–6, 2004.

[6] S. Visa and A. Ralescu, "Issues in mining imbalanced data sets-a review paper." in *Proceedings of the sixteen midwest artificial intelligence and cognitive science conference*, vol. 2005, April 2005, pp. 67–73.

[7] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, "A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches." *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 42, no. 4, pp. 463–484, 2012.

[8] V. López, A. Fernández, S. García, V. Palade, and F. Herrera, "An insight into classification with imbalanced data: Empirical results and

current trends on using data intrinsic characteristics." *Information Sciences*, no. 250, pp. 113–141.

[9] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique." *Journal of artificial intelligence research*, vol. 16, no. 1, pp. 321–357, 2002.

[10] G. E. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data." *ACM Sigkdd Explorations Newsletter*, vol. 6, no. 1, pp. 20–29, 2004.

[11] B. Zadrozny and C. Elkan, "Learning and making decisions when costs and probabilities are both unknown." in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, August 2001, pp. 204–213.

[12] P. Domingos, "Metacost: A general method for making classifiers cost-sensitive." in *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, August 1999, pp. 155–164.

[13] B. Zadrozny, J. Langford, and N. Abe, "Cost-sensitive learning by cost-proportionate example weighting." in *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*. IEEE, November 2003, pp. 435–442.

[14] L. Breiman, "Bagging predictors." *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.

[15] R. Tibshirani, "Regression shrinkage and selection via the lasso." *Journal of the Royal Statistical Society., Series B (Methodological)*, pp. 267–288, 1996.

[16] L. Breiman, "Random forests." *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[17] J. Davis and M. Goadrich, "The relationship between precision-recall and roc curves." in *Proceedings of the 23rd international conference on Machine learning*. ACM, June 2006, pp. 233–240.

SESSION

DATA SCIENCE AND DATA SERVICES

Chair(s)

Mahmoud Abou-Nasr
Peter Geczy
Robert Stahlbock
Gary M. Weiss

ISBN: 1-60132-431-6, CSREA Press ©

ISBN: 1-60132-431-6, CSREA Press ©

DL4MD: A Deep Learning Framework for Intelligent Malware Detection

William Hardy, Lingwei Chen, Shifu Hou, Yanfang Ye*, and Xin Li
Department of Computer Science and Electrical Engineering
West Virginia University, Morgantown, WV 26506, USA

Abstract- *In the Internet-age, malware poses a serious and evolving threat to security, making the detection of malware of utmost concern. Many research efforts have been conducted on intelligent malware detection by applying data mining and machine learning techniques. Though great results have been obtained with these methods, most of them are built on shallow learning architectures, which are still somewhat unsatisfying for malware detection problems. In this paper, based on the Windows Application Programming Interface (API) calls extracted from the Portable Executable (PE) files, we study how a deep learning architecture using the stacked AutoEncoders (SAEs) model can be designed for intelligent malware detection. The SAEs model performs as a greedy layerwise training operation for unsupervised feature learning, followed by supervised parameter fine-tuning (e.g., weights and offset vectors). To the best of our knowledge, this is the first work that deep learning using the SAEs model based on Windows API calls is investigated in malware detection for real industrial application. A comprehensive experimental study on a real and large sample collection from Comodo Cloud Security Center is performed to compare various malware detection approaches. Promising experimental results demonstrate that our proposed method can further improve the overall performance in malware detection compared with traditional shallow learning methods.*

Keywords: Malware Detection, Windows Application Programming Interface (API) Calls, Deep Learning.

1 Introduction

Malware is ***mal***icious soft***ware*** disseminated to infiltrate the secrecy, integrity, and functionality of a system [8], such as viruses, worms, trojans, backdoors, spyware. With computers and the Internet being essential in everyday life, malware poses a serious threat to their security. It is estimated that one in four computers operating in the U.S. are infected with malware [24]. The threat malware poses is not only emotional, but financial as well. According to a recent report from Kaspersky Lab, up to one billion dollars was stolen in roughly two years from financial institutions worldwide, due to malware attacks [16]. As a result, the detection of malware is of major concern to both the anti-malware industry and researchers.

In order to protect legitimate users from the attacks, the majority of anti-malware software products (e.g., Comodo, Symantec, Kaspersky) use the signature-based method of detection [10], [9]. Signature is a short string of bytes, which is unique for each known malware so that its future examples can be correctly classified with a small error rate [19]. However, this method can be easily evaded by malware attackers through the techniques such as encryption, polymorphism and obfuscation [29], [2]. Furthermore, malicious files are being disseminated at a rate of thousands per day [34], making it difficult for this signature-based method to be effective. In order to combat the malware attacks, intelligent malware detection techniques need to be investigated. As a result, many researches have been conducted on intelligent malware detection by applying data mining and machine learning techniques in recent years [27], [1], [23], [32], [35]. Based on different feature representations, different kinds of classification methods, such as Artificial Neural Network (ANNs), Support Vector Machine (SVM), Naïve Bayes (NB), and Decision Tree (DT), are used for model construction to detect malware [20], [7], [27]. Most of these methods are built on shallow learning architectures. Though they had isolated success in malware detection, shallow learning architectures are still somewhat unsatisfying for malware detection problems.

Deep learning (DL), a new frontier in data mining and machine learning, is starting to be leveraged in industrial and academic research for different applications (e.g., Computer Vision) [4], [14], [22]. A multilayer deep learning architecture is of superior ability in feature learning. More importantly, deep learning architectures overcome the learning difficulty through layerwise pretraining, i.e. pretraining multiple layers of feature detectors from the lowest level to the highest level to construct the final classification model [22]. This inspires us to devise a malware detection architecture based on deep learning.

In this paper, a deep learning architecture using the stacked AutoEncoders (SAEs) model for malware detection is studied, with the input resting on Windows Application Programming Interface (API) calls extracted from the Portable Executable (PE) files. The SAEs model employs a greedy layerwise training operation for unsupervised feature learning, followed by supervised parameter fine-tuning (e.g., weights and offset vectors). To the best of our knowledge, this is the first work investigating deep learning using the SAEs model resting on Windows API calls in malware detection for real industrial application. For validation, a comprehensive experimental study on a real and large sample collection from Comodo Cloud Security Center is performed to compare various malware detection approaches. The experimental results obtained demonstrate that our proposed method can further improve the overall performance in malware detection compared with traditional shallow learning

*Corresponding author

methods and that deep learning is a promising architecture for malware detection.

The rest of the paper is structured as follows. Section 2 presents the related work. Section 3 describes the overview of our malware detection system. Section 4 introduces our proposed deep learning approach for malware detection. Section 5 evaluates the performance of our proposed method in comparison with other alternative methods in malware detection. Finally, Section 6 concludes.

2 Related work

Signature-based method is widely used in anti-malware industry [10], [9]. However, malware authors can easily evade this signature-based method through techniques such as encryption, polymorphism and obfuscation [29]. Driven by the economic benefits, the quantity, diversity and sophistication of malware has significantly increased in recent years [34]. In order to effectively and automatically detect malware from the real and large daily sample collection, new, intelligent malware detection systems have been developed by applying data mining and machine learning techniques [31], [27], [20], [33], [28], [13]. These intelligent malware detection systems are varied in their uses of feature representations and classification methods. Naïve Bayes on the extracted strings and byte sequences was applied in [27], which claimed that Naïve Bayes classifier performed better than traditional signature-based method. Kolter et al. [20] focused on static analysis of the executable files and compared Naïve Bayes, Support Vector Machine and Decision Tree based on the n-grams. Ye et al. [33] proposed IMDS performing associative classification on Windows API calls extracted from executable files. Shah et al. [28] applied various feature selection algorithms to obtain the feature sets from PE files and used Artificial Neural Networks to detect new and unknown malware. Hou et al. [13] developed the intelligent malware detection system using cluster-oriented ensemble classifiers resting on the analysis of Windows API calls. Most of these existing researches are built on shallow learning architectures.

Due to its superior ability in feature learning through multilayer deep architecture [11], deep learning is feasible to learn higher level concepts based on the local feature representations [25]. As a result, researchers have paid much attention to deep learning methods in the domains of natural language processing, computer vision, etc. [18], [6]. In recent years, limited research efforts have been devoted to the malware detection using deep learning [21], [25], [15]. Ouellette et al. [25] extracted control flow graphs to present malware samples, and used a deep probabilistic model (sum-product network) to compare the similarities between the unknown file samples and those of representative sample features from known classes of malware. Jung et al. [15] used the features of header, tags, bytecode and API calls and utilized an ensemble learner consisting of different deep learning networks (e.g., deep feed-forward neural network, deep recurrent neural network) to classify the Adobe Flash file samples. Li et al. [21] proposed a hybrid malicious code detection approach on the basis of AutoEncoder and Deep Belief Network, where AutoEncoder was used to reduce the dimensionality of data, and a Deep Belief Network was applied to detect malicious code.

Different from the previous work, based on a real and large sample collection from Comodo Cloud Security Center, resting on the analysis of Windows API calls, we attempt to explore a deep learning architecture with the SAEs model to learn generic features of malware and thus to detect newly unknown malware.

3 System architecture

Our deep learning framework for malware detection (short for DL4MD) is performed on the analysis of Windows API calls generated from the collected PE files. The system consists of two major components: feature extractor, and deep learning based classifier, as illustrated in Figure 1.

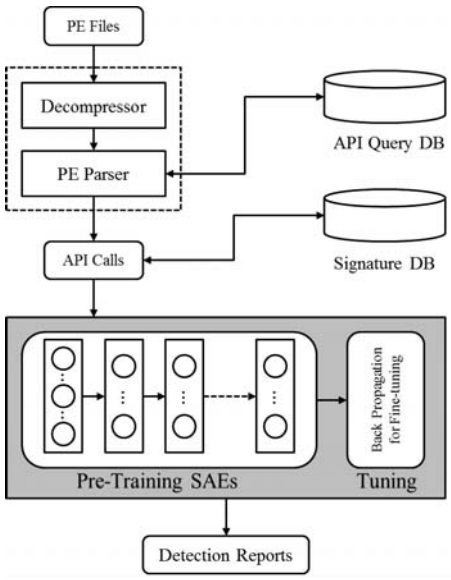


Fig. 1. DL4MD system architecture

- *Feature Extrator:* It is composed of Decompressor and PE Parser. The Decompressor needs to be employed first when a PE file is previously compressed by a binary compress tool (e.g., UPX and ASPack Shell) or embedded a homemade packer. The PE parser is used to extract the Windows API calls from each PE file. Through the API query database, the Windows API calls can be converted to a set of 32-bit global IDs representing the corresponding API functions (e.g., the API of "MAPI32.MAPIReadMail" is encoded as 0x00600F12). Then the API calls are stored as the signatures of the PE files in the signature database.
- *Deep Learning based Classifier:* Based on the Windows API calls, a deep learning architecture using SAEs model is designed to perform unsupervised feature learning, supervised fine-tuning, and thus malware detection. (See Section 4 for details.)

TABLE I
AN EXAMPLE DATASET OF SAMPLE FILES

File	Extracted API Calls	Feature Vector	Class
f_1	API_1, API_3	$< 1,0,1,0,0 >$	c_m
f_2	API_2, API_3, API_5	$< 0,1,1,0,1 >$	c_b
f_3	API_2, API_3, API_4, API_5	$< 0,1,1,1,1 >$	c_b
f_4	API_3	$< 0,0,1,0,0 >$	c_m
f_5	API_1, API_2, API_4	$< 1,1,0,1,0 >$	c_g

4 Proposed method

4.1 Problem definition

Resting on the analysis of Windows API calls, which can reflect the behavior of program code pieces [32] (e.g., the API "*GetModuleFileNameA*" in "*Kernel32.DLL*" can be used to retrieve the complete path of the file that contains the specified module of current process), a file f_i in our dataset D is denoted to be of the form $\langle A_{f_i}, C_{f_i} \rangle$, where A_{f_i} is the Windows API call feature vector of file f_i, and C_{f_i} is the class label of file f_i (where $C_{f_i} \in \{c_m, c_b, c_g\}$; c_m denotes malicious, c_b benign, and c_g unknown). Let d be the number of all extracted Windows API calls in the dataset D. The feature of each file can be represented by a binary feature vector:

$$A_{f_i} =< a_{i1}, a_{i2}, a_{i3}, ..., a_{id} >, \quad (1)$$

where $a_{ij} = \{0,1\}$ (i.e., if f_i contains API_j, $a_{ij} = 1$; otherwise, $a_{ij} = 0$). Table I shows a sample example.

Our malware detection problem statement can be stated as follows: *Given a dataset D as defined above, assign a label (i.e., c_m or c_b) to each unlabeled file (i.e., c_g) based on its feature vector.*

Although classification methods based on shallow learning architectures, such as Support Vector Machine (SVM), Naïve Bayes (NB), Decision Tree (DT), and Artificial Neural Network (ANN), can be used to solve the above malware detection problem, deep learning has been demonstrated to be one of the most promising architectures for its superior layerwise feature learning models and can thus achieve comparable or better performance [11]. Typical deep learning models include stacked AutoEncoders (SAEs), Deep Belief Networks with Restricted Boltzmann Machine, Convolutional Neural Networks [5], etc. In this paper, we explore a deep learning architecture with SAEs model for malware detection. The SAE model is a stack of AutoEncoders, which are used as building blocks to create a deep network.

4.2 AutoEncoder

An AutoEncoder, also called AutoAssociator, is an artificial neural network used for learning efficient codings [30]. Architecturally, the form of an AutoEncoder is with an input layer, an output layer and one or more hidden layers connecting them. The goal of an AutoEncoder is to encode a representation of the input layer into the hidden layer, which is then decoded into the output layer, yielding the same (or as close as possible) value as the input layer [4]. In this way, the hidden layer acts as another representation of the feature space, and in the case when the hidden layer is narrower (has fewer nodes) than the input/output layers, the activations of the final hidden layer can be regarded as a compressed representation of the input [4], [30], [5]. Figure 2 illustrates a one-layer AutoEncoder model with one input layer, one hidden layer, and one output layer.

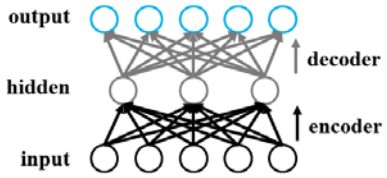


Fig. 2. A one-layer AutoEncoder model

Given a training set $\mathbf{X}$ of n samples $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, ..., \mathbf{x}_n\}$, where $\mathbf{x}_i$ ($i \in \{1, ..., n\}$) is a d_0-dimensional feature vector, i.e., $\mathbf{x}_i \in R^{d_0}$, and a sigmoid function denoted as

$$s(\mathbf{t}) = \frac{1}{1 + \exp^{-\mathbf{t}}}, \quad (2)$$

the AutoEncoder framework can be rigorously defined as follows [30], [22].

Encoder: In order to transform an input vector $\mathbf{x_i}$ into a hidden representation vector $\mathbf{y_i}$, the **encoder**, a deterministic mapping f_θ, is utilized. Its typical form is an affine mapping followed by a nonlinearity [30]:

$$\mathbf{y_i} = f_\theta(\mathbf{x_i}) = s(\mathbf{W}\mathbf{x_i} + \mathbf{b}), \quad (3)$$

where $\mathbf{W}$ is a $d_0 \times d_1$ weight matrix, d_1 is the number of hidden units, $\mathbf{b}$ is an offset vector of dimensionality d_1, and θ is the mapping parameter set $\theta = \{\mathbf{W}, \mathbf{b}\}$.

Decoder: The resulting hidden representation $\mathbf{y_i}$ is then mapped back to a reconstructed d_0-dimensional vector $\mathbf{z_i}$ in the input space, using the **decoder** $g_{\theta'}$. Its typical form is again an affine mapping optionally followed by a nonlinearity [30], i.e., either $\mathbf{z_i} = g_{\theta'}(\mathbf{x_i}) = \mathbf{W}'\mathbf{y_i} + \mathbf{b}'$ or

$$\mathbf{z_i} = g_{\theta'}(\mathbf{x_i}) = s(\mathbf{W}'\mathbf{y_i} + \mathbf{b}'), \quad (4)$$

where $\mathbf{W}'$ is a $d_1 \times d_0$ weight matrix, $\mathbf{b}'$ is also an offset vector of dimensionality d_0, and $\theta' = \{\mathbf{W}', \mathbf{b}'\}$.

Typically, the number of hidden units is much less than number of visible (input/output) ones ($d_1 < d_0$). As a result, when passing data through such a network, it first compresses (encodes) input vector to "fit" in a smaller representation, and then tries to reconstruct (decode) it back. The task of training is to minimize an error or reconstruction (using Equation 5), i.e. find the most efficient compact representation (encoding) for input data (Equation 6).

$$E(\mathbf{x}, \mathbf{z}) = \frac{1}{2} \sum_{i=1}^{n} ||\mathbf{x}_i - \mathbf{z}_i||^2. \quad (5)$$

$$\theta = \{\mathbf{W}, \mathbf{b}\} = \arg\min_{\theta} E(\mathbf{x}, \mathbf{z}). \quad (6)$$

Algorithm 1 illustrates the training procedure of AutoEncoder in our application. Note that other parameterized functions and loss functions can also be used for the encoder or decoder in the AutoEncoder framework [30].

Input: Data set $\mathbf{X}$ with n training samples: $\mathbf{x}_i =< A_{f_i}, C_{f_i} >$, where $i \in \{1, ..., n\}$ and $C_{f_i} \in \{c_m, c_b\}$
Output: Parameter set $\theta = \{\mathbf{W}, \mathbf{b}\}$

Initialize ($\mathbf{W}$, $\mathbf{b}$);
while *training error E hasn't converged or the designated iteration hasn't reached* **do**
  for *each input* $\mathbf{x}_i$ **do**
    Compute activations $\mathbf{y}_i$ at the hidden layer, and obtain an output $\mathbf{z}_i$ at the output layer;
  end
  Calculate the training error $E(\mathbf{x}, \mathbf{z})$;
  Backpropagate the error through the net and update parameter set $\theta = \{\mathbf{W}, \mathbf{b}\}$;
end

Algorithm 1: The algorithm for training AutoEncoder in malware detection

4.3 Deep learning architecture with SAEs

To form a deep network, an SAE model is created by daisy chaining AutoEncoders together, known as stacking: the output of one AutoEncoder in the current layer is used as the input of the AutoEncoder in the next [5]. More rigorously, with an SAE deep network with h layers, the first layer takes the input from the training dataset and is trained simply as an AutoEncoder. Then, after the k^{th} hidden layer is obtained, its output is used as the input of the $(k+1)^{th}$ hidden layer, which is trained similarly. Finally, the h^{th} layer's output is used as the output of the entire SAE model. In this manner, AutoEncoders can form a hierarchical stack. Figure 3 illustrates a SAEs model with h hidden layers.

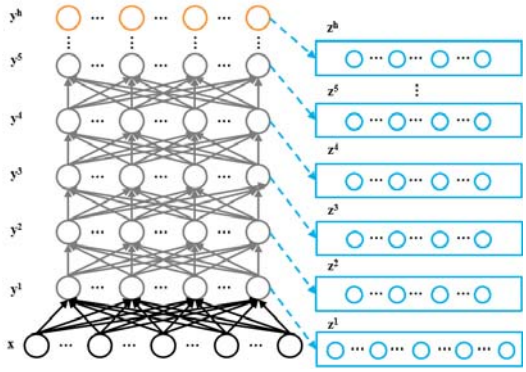


Fig. 3. A stacked AutoEncoders model

To use the SAEs for malware detection, a classifier needs to be added on the top layer. In our application, the SAEs and the classifier comprise the entire deep architecture model for malware detection, which is illustrated in Figure 4.

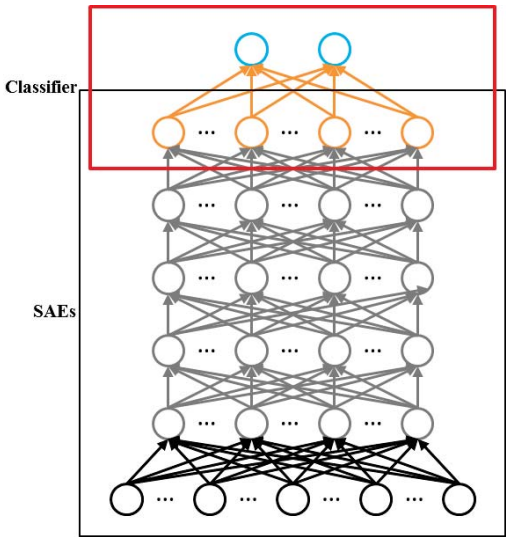


Fig. 4. Deep learning model for malware detection.

It is simple to train the deep network using Back Propagation (BP) with the gradient-based optimization technique, however, deep networks trained in this way are known to have poor performance. Fortunately, the greedy layerwise unsupervised learning algorithm developed by Hinton et al. [12] has overcome this problem. The key point of this algorithm is to pretrain the deep network layer by layer in a bottom-up manner and then fine-tune the parameters by applying BP in a top-down manner, which obtains better results. Resting on [12], [5], the training algorithm for malware detection using a deep learning architecture with SAEs is described in Algorithm 2.

Input: Data set $\mathbf{X}$ with n training samples: $\mathbf{x}_i =< A_{f_i}, C_{f_i} >$, where $i \in \{1, ..., n\}$ and $C_{f_i} \in \{c_m, c_b\}$; h: number of hidden layer; k_j: number of neurons for each layer, where $j \in \{1, ..., h\}$
Output: Parameter sets θs

for *each layer* $l (l \in \{1, ..., h\})$ *in SAEs* **do**
  Use Algorithm 1 to train the AutoEncoder at each layer;
end
Initialize ($\mathbf{W}^{h+1}$, $\mathbf{b}^{h+1}$) at the classifier layer;
Calculate the labels for each training sample $\mathbf{x}_i$;
Perform BP in a supervised way to tune the parameter sets θs of all layers;

Algorithm 2: The algorithm for training the deep learning network with SAEs in malware detection

5 Experimental results and analysis

In this section, we conduct two sets of experiments based on a real and large sample collection obtained from Comodo Cloud Security Center to fully evaluate the performance of our deep learning model in malware detection: (1) In the first

TABLE II
PERFORMANCE MEASURES IN MALWARE DETECTION

Measure	Description
TP	Number of files correctly classified as malicious
TN	Number of files correctly classified as benign
FP	Number of files mistakenly classified as malicious
FN	Number of files mistakenly classified as benign
TP Rate (TPR)	$\frac{TP}{TP+FN}$
FP Rate (FPR)	$\frac{FP}{FP+TN}$
Accuracy (ACY)	$\frac{TP+TN}{TP+TN+FP+FN}$

Remark: TP: True Positive, TN: True Negative, FP: False Positive, and FN: False Negative.

set of experiments, we evaluate the deep learning networks with different parameters (i.e., different numbers of hidden layers and neurons); (2) In the second set of experiments, we conduct a comparison between our proposed method and other shallow learning based classification methods (i.e., ANN, SVM, NB, and DT).

5.1 Experimental setup

The dataset obtained from Comodo Cloud Security Center contains $50,000$ file samples, where $22,500$ are malware, $22,500$ are benign files, and $5,000$ are unknown (with the analysis by the anti-malware experts of Comodo Security Lab, $2,500$ of them are labeled as malware and $2,500$ of them are benign). In our experiments, those $45,000$ file samples are used for training, while the $5,000$ unknown files are used for testing. $9,649$ Windows API calls are extracted from these $50,000$ file samples, so all the file samples can be represented as binary feature vectors with $9,649$-dimensions (described in Section 4.1). To quantitatively validate the experimental results, we use the performance measures shown in Table II. All experiments are conducted in the environment: 64 Bit Windows 8.1 on an Intel (R) Core (TM) i7-4790 Processor (3.60GHz) with 16GB of RAM, using MySQL and C++.

5.2 Evaluations of different deep networks

In this section, based on the dataset described in Section 5.1, we evaluate the deep learning networks with different parameters (i.e., different numbers of hidden layers and neurons). The results in Table III and Figure 5 demonstrate the effectiveness of our proposed algorithm in malware detection. The deep learning model with 3 hidden layers and 100 neurons at each hidden layer is superior to other deep networks in training and testing accuracy for malware detection.

5.3 Comparisons between deep learning and other shallow learning based classification methods

In this section, using the same dataset described in Section 5.1, we conduct a comparison between our proposed

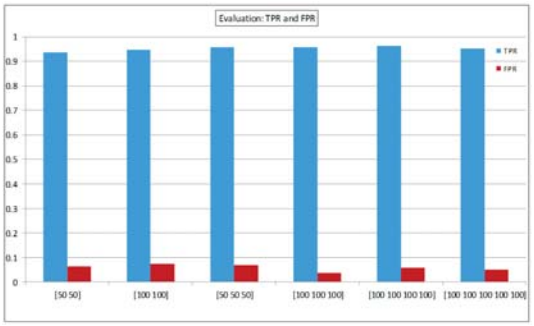


Fig. 5. Comparisons of testing TPR and FPR between different deep networks

deep learning framework (DL4MD) and other shallow learning based classification methods (i.e., Artificial Neural Network (ANN), Support Vector Machine (SVM), Naïve Bayes (NB), and Decision Tree (DT)) in malware detection. The results in Table IV, Figure 6 and Figure 7 show that our proposed deep learning framework (DL4MD) outperform ANN, SVM, NB, and DT in malware detection.

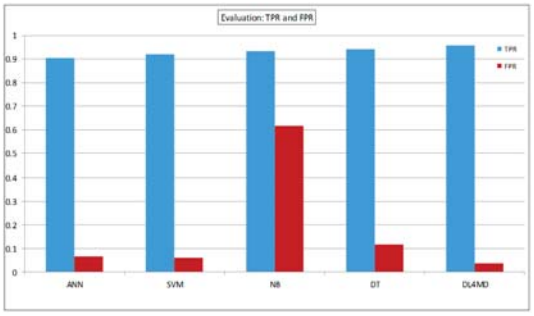


Fig. 6. Comparisons of testing TPR and FPR between ANN, SVM, NB, DT, and DL4MD

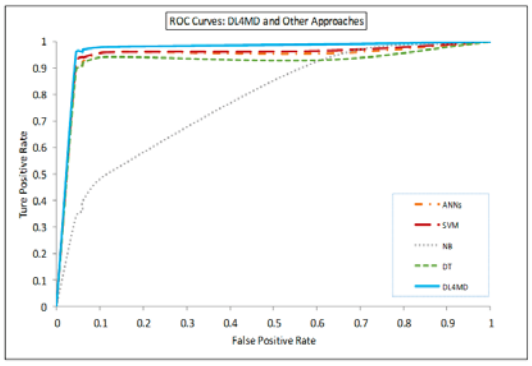


Fig. 7. Comparisons of ROC curves between ANN, SVM, NB, DT, and DL4MD

Our proposed deep learning framework (DL4MD) also performs well in detection efficiency: it just takes about 0.1 second for each unknown sample prediction, including feature extraction. This makes our system a practical solution for intelligent malware detection in real industrial application.

TABLE III
EVALUATION OF DIFFERENT DEEP NETWORKS

Training

Hidden Layers	Neurons	TP	FP	TN	FN	ACY
2	[50 50]	21,859	1,434	21,066	641	0.9539
2	[100 100]	22,142	1,460	21,040	358	0.9596
3	[50 50 50]	22,110	1,295	21,205	390	0.9626
3	**[100 100 100]**	**22,035**	**953**	**21,547**	**465**	**0.9685**
4	[100 100 100 100]	22,150	1,178	21,322	350	0.9660
5	[100 100 100 100 100]	22,055	977	21,523	445	0.9684

Testing

Hidden Layers	Neurons	TP	FP	TN	FN	ACY
2	[50 50]	2,368	161	2,339	132	0.9414
2	[100 100]	2,391	185	2,315	109	0.9412
3	[50 50 50]	2,396	170	2,330	104	0.9452
3	**[100 100 100]**	**2,396**	**114**	**2,386**	**104**	**0.9564**
4	[100 100 100 100]	2,408	147	2,353	92	0.9550
5	[100 100 100 100 100]	2,384	128	2,372	116	0.9512

TABLE IV
COMPARISONS BETWEEN DEEP LEARNING AND OTHER SHALLOW LEARNING BASED CLASSIFICATION METHODS

Training

Method	TP	FP	TN	FN	ACY
ANN	21,338	1,781	20,719	1,162	0.9346
SVM	21,610	1,576	20,924	890	0.9452
NB	21,532	9,940	12,560	968	0.7576
DT	21,630	1,560	20,940	870	0.9460
DL4MD	**22,035**	**953**	**21,547**	**465**	**0.9685**

Testing

Method	TP	FP	TN	FN	ACY
ANN	2,264	163	2,337	236	0.9202
SVM	2,305	152	2,348	195	0.9306
NB	2,332	1,541	959	168	0.6582
DT	2,357	292	2,208	143	0.9130
DL4MD	**2,396**	**114**	**2,386**	**104**	**0.9564**

6 Conclusion and future work

In this paper, based on Windows API calls extracted from a real and large file sample collection, we design a deep learning framework using the SAEs model for malware detection, which consists of two phases: *unsupervised pre-training* and *supervised backpropagation*. The SAEs model performs as unsupervised pretraining in a greedy layer-wise fashion putting the parameters of all layers in a region of parameter space in a bottom-up way, and then supervised BP is adopted to tune the multilayer model's parameters in a top-down direction. To the best of our knowledge, this is the first work investigating deep learning using the SAEs model based on Windows API calls in malware detection for real industrial application. A comprehensive experimental study on a real and large file collection from Comodo Cloud Security Center is performed to compare various malware detection approaches. The experimental results obtained demonstrate that our proposed method can further improve the overall performance in malware detection compared with traditional shallow learning methods and that deep learning is a viable architecture for malware detection.

In our future work, we will further explore how sparsity constraints are imposed on AutoEncoder and how sparse SAEs can be designed to further improve malware detection effectiveness. Meanwhile, it would be interesting to investigate other deep learning models for malware detection.

Acknowledgment

The authors would also like to thank the anti-malware experts of Comodo Security Lab for the data collection as well as helpful discussions and supports. This work is partially supported by the U.S. National Science Foundation under grant CNS-1618629.

References

[1] M. Bailey, J. Oberheide, J. Andersen, Z. Mao, F. Ahanian, and J. Nazario. Automated classification and analysis of internet malware. In *RAID 2007, LNCS*, 178-197, 2007.

[2] P. Beaucamps, and E. Filiol. On the possibility of practically obfuscating programs towards a unified perspective of code protection. In *Journal in Computer Virology*, 3 (1), 2007.

[3] Y. Bengio, and Y. LeCun. Scaling Learning Algorithms towards AI. In *Large-scale kernel machines*, 34(5), 2007.

[4] Y. Bengio. Learning Deep Architectures for AI. In *Foundations and Trends in Machine Learning*, Vol 2(1), 1-127, 2009.

[5] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. Greedy Layer-Wise Training of Deep Networks. In *Advances in Neural Information Processing Systems 19 (NIPS'06)*, 153-160, 2007.

[6] R. Collobert, and J. Weston. A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In *Proceedings of 25th ICML*, 160-167, 2008.

[7] R. A. Dunne. A Statistical Approach to Neural Networks for Pattern Recognition. In *Wiley-Interscience, 1st edition*, 2007.

[8] M. Egele, T. Scholte, E. Kirda, and C. Kruegel. A Survey on Automated Dynamic Malware Analysis Techniques and Tools. In *ACM CSUR*, Vol 44(2), 6:1-6:42, 2008.

[9] E. Filiol. Malware pattern scanning schemes secure against blackbox analysis. In *J. Comput. Virol*, Vol 2(1), 35-50, 2006.

[10] E. Filiol, G. Jacob, and M. L. Liard. Evaluation methodology and theoretical model for antiviral behavioural detection strategies. In *J. Comput. Virol*, Vol 3(1), 27-37, 2007.

[11] G. E. Hinton, and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. In *Science*, Vol 313(5786), 504-507, 2006.

[12] G. E. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. In *Neural Computation*, Vol 18, 1527-1554, 2006.

[13] S. Hou, L. Chen, E. Tas, I. Demihovskiy, and Y. Ye. Cluster-Oriented Ensemble Classifiers for Malware Detection. In *IEEE International Conference on Sematic Computing (IEEE ICSC)*, 189-196, 2015.

[14] W. Huang, G. Song, H. Hong, and K. Xie. Deep Architecture for Traffic Flow Prediction: Deep Belief Networks With Multitask Learning. In *IEEE Transactions on Intelligent Transportation Systems*, Vol 15(5), 2191-2201, 2014.

[15] W. Jung, S. Kim, and S. Choi. Poster: Deep Learning for Zero-day Flash Malware Detection. In *36th IEEE Symposium on Security and Privacy*, 2015.

[16] Kaspersky Lab. The Great Bank Robbery. In *http://www.kaspersky.com/about/news/virus/2015/Carbanak-cybergang-steals-1-bn-USD-from-100-financial-institutions-worldwide*, 2015.

[17] Kaspersky Lab. Kaspersky Anti-Virus SDK v.8 Core Detection Technologies. In *White Paper*, August, 2009.

[18] K. Kavukcuoglu, P. Sermanet, Y. Boureau, K. Gregor, M. Mathieu, and Y. LeCun. Learning Convolutional Feature Hierachies for Visual Recognition. In *Advances in Neural Information Processing Systems (NIPS 2010)*, 23, 2010.

[19] J. Kephart, and W. Arnold. Automatic extraction of computer virus signatures. In *Proceedings of 4th Virus Bulletin International Conference*, 178-184, 1994.

[20] J. Kolter, and M. Maloof. Learning to detect malicious executables in the wild. In *SIGKDD*, 2004.

[21] Y. Li, R. Ma, and R. Jiao. A Hybrid Malicious Code Detection Method based on Deep Learning. In *International Journal of Security and Its Applications*, Vol 9(5), 205-216, 2015.

[22] Y. Lv, Y. Duan, W. Kang, Z. Li, and F. Wang. Traffic Flow Prediction With Big Data: A Deep Learning Approach. In *IEEE Transactions on Intelligent Transportation Systems*, Vol 16(2), 865-873, 2015.

[23] M. M. Masud, T. M. Al-Khateeb, K. W. Hamlen, J. Gao, L. Khan, J. Han, and B. Thuraisingham. Cloud-Based Malware Detection for Evolving Data Streams. In *ACM TMIS*, Vol 2(3), 16:1-16:27, 2008.

[24] Organisation for Economic Co-operation and Development. Malicious software (malware):Asecurity threat to the internet economy. White Paper, June, 2008.

[25] J. Ouellette, A. Pfeffer, and A. Lakhotia. Countering Malware Evolution Using Cloud-Based Learning. In *8th International Conference on Malicious and Unwanted Software: "The Americas" (MALWARE)*, 85-94, 2013.

[26] Y. Park, Q. Zhang, D. Reeves, and V. Mulukutla. AntiBot: Clustering Common Semantic Patterns for Bot Detection. In *IEEE 34th Annual Computer Software and Applications Conference*, 262-272, 2010.

[27] M. Schultz, E. Eskin, and E. Zadok. Data mining methods for detection of new malicious executables. In *Proccedings of IEEE Symposium on Security and Privacy*, 2001.

[28] S. Shah, H. Jani, S. Shetty, and K. Bhowmick. Virus Detection using Artificial Neural Networks. In *International Journal of Computer Applications*, vol. 84(5), 2013.

[29] A. Sung, J. Xu, P. Chavez, and S. Mukkamala. Static analyzer of vicious executables (save). In *Proceedings of the 20th ACSAC*, 326-334, 2004.

[30] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P. Manzagol. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. In *Journal of Machine Learning Research*, Vol 11, 3371-3408, 2010.

[31] J. Wang, P. Deng, Y. Fan, L. Jaw, and Y. Liu. Virus detection using data mining techniques. In *Proccedings of ICDM03*, 2003.

[32] Y. Ye, D. Wang, T. Li, D. Ye, and Q. Jiang. An intelligent PE-malware detection system based on association mining. In *Journal in Computer Virology*, Vol 4, 323-334, 2008.

[33] Y. Ye, D. Wang, T. Li, and D. Ye. IMDS: Intelligent Malware Detection System. In *Proceedings of the 13th ACM SIGKDD*, 1043-1047, 2007.

[34] Y. Ye, T. Li, S. Zhu, W. Zhuang, E. Tas, U. Gupta, and M. Abdulhayoglu. Combining File Content and File Relations for Cloud Based Malware Detection. In *Proceedings of ACM International Conference on Knowledge Discovery and Data Mining (ACM SIGKDD)*, 222-230, 2011.

[35] Y. Fan, Y. Ye and L. Chen. Malicious sequential pattern mining for automatic malware detection. In *Expert Systems with Applications*, Vol 52, 16-25, 2016.

Skill Identification Using Time Series Data Mining

Toshiyuki MAEDA[1] and Masumi YAJIMA[2]
[1] *Faculty of Management Information, Hannan University, Japan*
[2] *Faculty of Economics, Meikai University, Japan*

Keywords: Time Series Data, Sports Skill, Data Mining, Motion Picture, Knowledge Acquisition

Abstract: This paper addresses sports skill identification using time series motion picture data, focused on volleyball. In this paper, volleyball play is analyzed with motion picture data recorded by hi-speed cam-coder, where we do not use physical information such as body skeleton model, and so on. Time series data are obtained from the motion picture data with marking points, and analyzed using data mining methods such as Naive Bayes, and other tree learning algorithm. We attempt to identify technical skill models of volleyball attacks.

1 Introduction

For not only engineering skills but also sports skills, many researchers treat body structure models and/or skeleton structure models obtained from physical information such as activity or biomechatronical data for sports skill researches [1].

Those might be because those researchers believe that internal models of technical skill are structured physically, with some skill levels which are human intention, environmental adjustment, and so on [2].

For instance, Matsumoto and others describes skilled workers skills which have actually internal models of structured skill architecture and they choose an action process from internal models adjusted with environment [3]. It is even though hard for skilled workers to represent internal models by themselves. They reflect involuntarily their own represented actions, and achieve highly technical skills with internal models.

We had, however, researched that fore-hand strokes of table tennis play exemplify sports action, and classify skill models using motion picture data analysis without body structure model nor skeleton structure model. We had evaluated those into three play levels as expert/intermediate/novice, and classify the models using data mining technologies [4, 5]. That means this research is a challenge to clarify internal models only from represented image data.

We hence have an attempt to apply our research framework to other sports skills, and then this paper addresses a personal sports skill identification using time series motion picture data, focused on volleyball.

2 Related Works

Wilkinson[6] describes that qualitative skill analysis is an essential analytic tool for physical educators and refers to a process in which a teacher identifies discrepancies between the actual response observed and the desired response. Providing instruction for preserving teachers regarding how to recognize errors has been largely neglected in teacher preparation. The purpose of this study was to evaluate an alternative approach for teaching qualitative skill analysis to undergraduates. The study evaluated the effectiveness of a visual-discrimination training program. The subjects were 18 undergraduate students. The visual-discrimination training program was introduced using a multiple-baseline design across three volleyball skills: the forearm pass, the overhead pass, and the overhead serve. After the introduction of each instructional component, subjects made abrupt improvements in correctly analyzing the volleyball skill. This approach for teaching qualitative skill analysis is one alternative to the conventional techniques currently being used in professional preparation.

Watanabe et al.[7] shows a method for the measurement of sports form. The data obtained can be used for quantitative sports-skill evaluation. Here, they focus on the golf-driver-swing form, which is difficult to measure and also difficult to improve. The measurement method presented was derived by kinematic human-body model analysis. The system was developed using three-dimensional (3-D) rate gyro sensors set of positions on the body that express the 3-D rotations and translations during the golf swing. The system accurately measures the golf-driver-swing form of golfers. Data obtained by this system can be

related quantitatively to skill criteria as expressed in respected golf lesson textbooks. Quantitative data for criteria geared toward a novice golfer and a mid-level player are equally useful.

Barzouka et al.[8] examine the effect of feedback with simultaneous skilled model observation and self-modeling on volleyball skill acquisition. 53 pupils 12 to 15 years old formed two experimental groups and one control group who followed an intervention program with 12 practice sessions for acquisition and retention of how to receive a ball. Groups received different types of feedback before and in the middle of each practice session. Reception performance outcome (score) and technique in every group were assessed before and at the end of the intervention program and during the retention phase. A 3 (Group) $\times$ 3 (Measurement Period) multivariate analysis of variance with repeated measures was applied to investigate differences. Results showed equivalent improvement in all three groups at the end of the intervention program. In conclusion, types of augmented feedback from the physical education teacher are effective in acquisition and retention of the skill for reception in volleyball.

3 Experiments

Our research is to identify internal models from observed motion picture data and skill evaluation with represented actions, without measurement of the body structure or the skeleton structure.

We focus on volleyball attack among various sports, and analyze volleyball skills of stacks from observed motion picture data and skill evaluation with represented actions.

For the feasibility research, we have recorded motion pictures of 6 subjects who are 3 expert / 3 novice-level university students. As skill evaluation of representing action, We classify the levels as follows;

- Expert class: members of volleyball club at university,
- Novice class: inexperienced students.

Figure 1 shows positions of marking setting. Each player is marked at 4 points on the right arm as;

1. Left knee,
2. Right waist,
3. Right shoulder, and
4. Right elbow.

We have recorded swing traces of stacks using a high-speed cam-corder (resolution: 512 $\times$ 384 pixel and frame-rate: 300 frames per seconds) installed besides of the players. On playing in 5 minutes, several attack motions are recorded for each player.

4 Skill Identification

From the recorded motion pictures, 100 to 200 frames are retrieved from the beginning of take-back to the ball until the end of the attack. We have then distributed two dimensional axes positions (pixel values) of 4 marking points for each frame, where the starting point is set at the shoulder position of the first frame.

We then attempt an investigation using data mining technique. The skill evaluation of representing action consists of two classes such as Expert and Novice. Each marking position is represented two dimensional and so the observed data are reconstructed in 48-input / 2-class output.

For applying observed data of fore-hand strokes of 6 subject players, we reconstruct time series data from the original data. One datum is a set of 48-tuple numbers such as 4 markings $\times$ 2 axis $(x,y) \times 6$ frames, and each datum is overlapped with 3 frames data (from fourth to sixth frame) of the next datum for presenting linkage of each datum (see Figure 2).

We use an integrated data mining environment "weka" [9] and analyze the data by analyzing methods of J48 (an implementation of C4.5) and NBT (Naive Bayes Tree).

4.1 Pre-examination

We have had a pre-examination for applying our research framework into volleyball skills. Table 1 shows the recognition rate of the data sets. As for expert players, two third of data on three players are used as learning data, and the rest for evaluation.

Table 1: Recognition rate of modified data sets.

	Recognition Rate(%)	
	Cross validation	Learning and test
J48	95.9	48.2
NBT	97.7	62.7

Table 2 shows the discrimination of classes of NBT classification.

In those results, the recognition rates for evaluation data are not so good, though NBT makes better results for evaluation data. On the contrary, the result of the number of class recognition for each method in Table 2 implies that NBT tend to recognize Expert as Novice. This implies that, even in Expert class, data

FIGURE 1: Measurement markings.

Table 2: Discrimination of classes for NBT classification.

	Classified as	
	Expert	Novice
Expert	66	58
Novice	14	55

Table 3: Recognition rate of personal data sets.

	Recognition Rate(%)	
	Cross validation	Learning and test
One Expert and one Novice	96.9	97.5
One Expert and another Expert	94.2	100.0

may have some variation. We then focus on personal identification, as data variation may cause from personal skill variation e.g. an expert position in plays such as attacker, center player, and so on.

4.2 Personal Identification

As mentioned above, we here analyze for personal identification, where two subjects are compared for each test. One test is with an Expert and a Novice, and the other test is with two Experts. We here use NBT for these tests as NBT are better than J48 on above tests.

These results are fairly good and suggest this analysis process can identify each person, especially identification for two Experts, though we need further investigation.

5 Conclusion

This paper addresses analysis and classification for internal models for technical skills as evaluation skillfulness for volleyball attack motion, and discuss skill identification. We had some experiments and

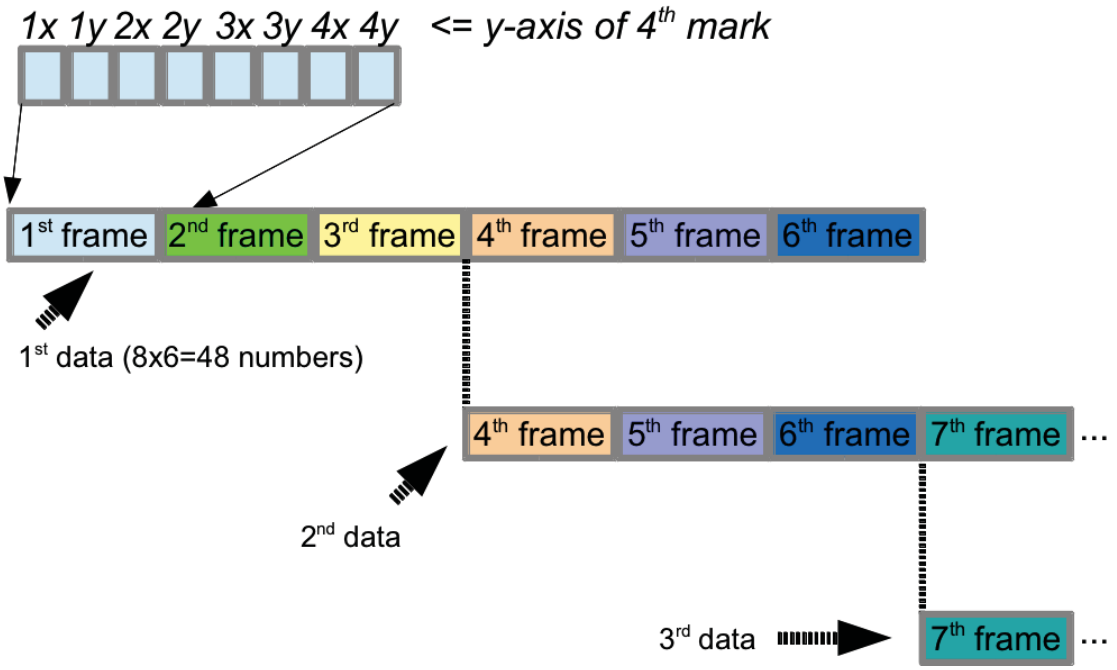


FIGURE 2: Data structure from isolated pictures.

some results imply that expert or intermediate players can make some categorical groups for technical skills, but there seems not to be a category for novice players because of various individual technical skills. Furthermore, for applying observed data of volleyball stacks of players, we reconstruct time series data from the original data and analyze the new data by data mining techniques such as J48, NBT, where the recognition rate for evaluation data is fairly good, and NBT makes better results for learning and evaluation data. Personal analysis furthermore may be better categorized. As future plans, we have to progress further experiments, and measure more precise data and then analyze if needed.

Acknowledgment

Part of this research was supported by JSPS KAKENHI Grant Number 15K02185 and 15K03802. This research was assisted, especially for data management, by Mr. Y. Tamari and Mr. Y. Tsujino at Hannan University. The authors greatly appreciate those.

References

[1] Y. Mochizuki, R. Himeno, and K. Omura: Artificial Skill and a New Principle in Sports (Special Issue on Digital Human : Measurement and Modeling of Human Functions). (in Japanese) *System, Control, and Information*, Vol.46, No.8, pp.498–505 (2002)

[2] T. Shiose, T. Sawaragi, K. Kawakami, and O. Katai: Technological Scheme for the Preservation of Technique from Ecological Psychological Approach. (in Japanese), *Ecological Psychology Research*, Vol.1, No.1, pp.11–18 (2004)

[3] Y. Matsumoto: *Organization and Skill – Organization Theory of Preservation of Technique* (in Japanese), Hakuto Shobo (2003)

[4] T. Maeda, I. Hayashi, and M. Fujii: Time Series Data Analysis for Sport Skill. *Proceedings of the 2012 12th International Conference on Intelligent Systems Design and Applications* , pp.392–397, Kochi, India (2012)

[5] T. Maeda, I. Hayashi, M. Fujii and T. Tasaka: Sport Skill Classification Using Time Series Motion Picture Data. *Proceedings of thethe 40th Annual Conference of the IEEE Industrial Electronics Society (IECON 2014)* , pp.5272–5277, Dallas, TX USA (2014)

[6] S. Wilkinson: A Training Program for Improving Undergraduates' Analytic Skill in Volleyball. *Journal of Teaching in Physical Education*, Vol.11, Iss.2, pp.177–194 (1992)

[7] K. Watanabe, and M. Hokari: Kinematical analysis and measurement of sports form. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, Vol.36, Iss.3, pp.549–557 (2006)

[8] K. Barzouka, N. Bergeles, and D. Hatziharitos: EFFECT OF SIMULTANEOUS MODEL OBSERVATION AND SELF-MODELING OF VOLLEYBALL SKILL ACQUISITION. *Perceptual and Motor Skills*, Vol.104, pp.32–42 (2007)

[9] http://www.cs.waikato.ac.nz/ml/weka/ (2014)

Internet of Things Technologies to Rationalize the Data Acquisition in Industrial Asset Management

Sini-Kaisu Kinnunen[1], Antti Ylä-Kujala[1], Salla Marttonen-Arola[1], Timo Kärri[1], David Baglee[2]
[1]School of Business and Management, Lappeenranta University of Technology, Lappeenranta, Finland
[2]Department of Computing, Engineering and Technology, University of Sunderland, Sunderland, UK

Abstract - *Based upon the needs of industry, academia is producing a growing number of decision-making models and tools for maintenance and asset management. At the same time the amount of data available for companies is overwhelming and unorganized, as our previous research and experience have shown. The acquisition of relevant input data for these models can be time consuming, prone to error and often difficult with the current technologies and systems. The emerging Internet of Things (IoT) technologies could rationalize data processes from acquisition to decision making if future research is focused on the exact needs of industry. This paper contributes to this field by examining and categorizing the applications available through IoT technologies in the management of industrial asset groups. Previous literature and a number of industrial experts are used to identify the feasibility of IoT technologies in asset management. This paper describes a preliminary study, which highlights the research potential of specific IoT technologies, for further research related to smart factories of the future.*

Keywords: Internet of Things, IoT, Data acquisition, Asset management, Smart factory

1 Introduction

As a result of the rapid and world-wide globalization in the industry today, companies and other organizations are networking, whether intentionally or unintentionally, in increasing pace. Complex interdependencies in the organizational interface set entirely new requirements for data acquisition and data transmission as well as for generating usable decision-making information from the data. Based upon the above-mentioned need to manage and control inter-organizational environments, academia is producing a growing number of decision-making models and tools designed for industrial asset management in particular. Our contributions in this area are the "Life-Cycle Model for Maintenance Service Management", created for inter-organizational operation and maintenance planning and decision making of a production asset [1]–[2], and the "Flexible Asset Management Model" or "FAM-model", targeted for optimizing a network's asset quantities and balance sheet -positioning in a strategic level [3].

However, the amount of data in companies and numerous information systems are constantly growing, which has caused problems in separating relevant data from irrelevant data. Therefore, it has proven very difficult to generate accurate, adequate and timely data for industrial asset management models and tools, such as the "Life-Cycle Model" or the "FAM-model". One potentially viable solution to improved data acquisition and transmission are Internet of Things (IoT) technologies that will automate asset-related data processes in smart factories of the future through embedded communication within the existing internet infrastructure [4]. IoT does not however intrinsically solve any difficulties in data utilization, i.e. turning data to business information, where suitable data penetration and analytics software or techniques, so-called middleware is highlighted instead [5]. As IoT technologies are altogether a novel approach, the field remains somewhat unclear, which creates a need to carry out research especially from an industrial asset management perspective. Therefore research is needed to clarify feasible industrial applications in order to improve data exploitation and data-based decision making in industrial environment. The objective of this paper is finding and studying asset management-wise relevant Internet of Things technologies by connecting an industrial asset group to an IoT technology both in theoretical and in practical level. Consequently, the objective of this paper is shaped into following research questions:

1. What are the essential IoT technologies to be employed in the data acquisition and data transmission of various physical industrial assets in smart factories of the future?

2. How does an industrial expert panel foresee the industrial asset management potential of IoT technologies in relation to their companies' businesses?

Our research employs two methods. Firstly, current knowledge on existing IoT technologies and their potential applications are studied by conducting a comprehensive literature review in order to achieve a theoretical overall view and determine if there is a research gap. Secondly, the outlook of industry, and therefore our empirical evidence, is mapped via an industrial expert panel that is comprised of industrial asset management and industrial maintenance specialists representing internationally distinguished companies in Finland and Sweden.

2 Theoretical framework

2.1 IoT technologies

IoT comprises of a network of smart objects that are connected to the Internet. In the context of industry, the term of Industrial Internet and Industry 4.0 are often used alongside IoT. Applications utilizing IoT technologies are increasing as enabling technologies are developing and becoming less expensive. IoT is transforming businesses and it has been stated to be an industry revolution taking place right now [6]. Companies are developing new applications and innovative uses for IoT technologies. IoT technologies have been applied to numerous environments, such as logistics, manufacturing, security, and healthcare. Hence, the applications vary from inventory control to e-Health applications. The possibilities of IoT enabling technologies and applications have received attention in the literature [7]–[10]. The categorization of IoT technologies is not uniform and different technologies are often applied together. Commonly discussed technologies are: RFID (Radio Frequency Identification), WSN (Wireless Sensor Networks), WSAN (Wireless Sensor and Actuator Networks), WPAN (Wireless Personal Area Networks), NFC (Near Field Communication), as well as naming and localization technologies. These technologies are mainly used for identification, sensing and communication purposes.

RFID technology uses radio waves to identify items. RFID technology is primary used for identification purposes but it enables also to store limited data in RFID tags [11]. RFID technology consists of electronic tags (RFID tags) and RFID readers. RFID tag stores the unique code of the attached object and RFID reader can act as a gateway to the Internet by transmitting the object identity, read-time and the object location which therefore enables real-time tracking [12]. RFID technology enables to automate the process of object identification and eliminate human link. Reference [13] has reviewed applications based on RFID technology in different industries. RFID technology enables to reduce shrinkage, prevent stock outs and excess stocks, improve data accuracy, and increase information visibility in the supply chain. RFID technology is applied, for example, to inventory control, product tracking, building access control and real-time location system in complex manufacturing processes [13].

WSN technology refers to a group of sensors that can monitor and record the physical conditions of the environment. WSN technology utilizes sensors to collect data about the targeted phenomenon and transmits the data to base stations that can be connected to the Internet. [12] There are many different types of sensors and they are able to monitor a wide variety of physical conditions, including temperature, humidity, vehicular movement, pressure, noise levels and mechanical stress levels on attached objects [14]. WSN technology can be applied to great variety of different applications, for instance, to machinery condition monitoring, traffic estimation, and power consumption monitoring [11], [15]–[16].

WSAN technology combines sensing technologies with actuating possibilities. Sensors gather information about the physical world, while actuators take decisions and then perform appropriate actions upon the environment. This allows remote and automated interaction with environment. [17] WSAN technology provides innovative application possibilities and it is applied to variety of building automation applications, for example, to temperature control, to air conditioning system which reacts to the presence of people, and to other applications that can provide energy savings in buildings [18]–[20].

WPAN enables the energy efficient wireless access and data transfer to smart objects over a short distance (1 m–100 m). Examples of WPANs are Bluetooth and ZigBee. Bluetooth defines a complete WPAN architecture, including a security layer. However, the main disadvantage of the Bluetooth is its relatively high energy consumption. Therefore, Bluetooth is not usually used by sensors that are powered by a battery. ZigBee is developed to be corresponding technology but simpler and less expensive than Bluetooth. Additionally, ZigBee has low power consumption and is more energy efficient. [12], [21]

NFC refers to short-range high frequency wireless communication technology which enables the exchange of data between devices over a distance of less than 20 cm. NFC technology is compatible with existing smartcards and readers but also with other NFC devices and it is suitable for use in mobile phones [12]. NFC technology can be utilized in public transportation, proximity payment and access keys for offices and houses, for example. NFC utilized in mobile phones enables peer-to-peer communication between two NFC devices and, for instance, business cards and other information can be exchanged [12], [22].

In order to be able to communicate via the Internet, smart objects need appropriate naming technologies. Smart objects need to be identified and the access path to the object needs to be established. Examples of naming schemes are barcodes, 2D (two-dimensional) barcodes, EPC (Electronic Product Code) and IP (Internet Protocol) address. Naming technologies make possible to identify items with appropriate accuracy. Then, sometimes it is adequate to identify items at item group level and sometimes an item specific identification is needed. Item specific identification is needed especially when particular object or device needs to be accessed, for example when tracking vehicle [23] or controlling particular appliance in building automation [18]. Then EPC or IP address would be more appropriate choice.

Localization technologies can be categorized into three groups based on the infrastructure of technology: satellite based, mobile networks based and local area networks based technologies. Satellite positioning technology utilizes distance measurement to satellites to determine three-dimensional location. An example of satellite positioning systems is GPS

(Global Positioning System). A satellite positioning system is intended to be used outdoors and it is not suitable for indoors. Network based positioning technology is based on mobile networks that maintain the location data. Whereas, local area networks can utilize technologies, such as RF (Radio Frequency) signals or local positioning properties of Bluetooth [24].

By applying and combining these IoT technologies, there is a huge potential to develop enormous amount of new IoT based applications in different environments. When considering supporting technologies, such as mobiles and memory capabilities of cloud computing, the IoT based application possibilities increase. These IoT applications produce increasing amount of data which is known as big data. Big data is often semi-structured or unstructured by nature and collected data is useful only if it is analyzed [25]. The challenge is how to effectively exploit the collected data in applications and therefore turn data into business information. In order that companies actually start gathering, analyzing and using the data, the decision making value and potential end use of data must be transparent for them.

2.2 Definition of assets

An asset is generally defined by ISO 55000 standard [26] as "an item, thing or entity that has potential or actual value to an organization". The value will vary between different organizations and their stakeholders, and can be tangible or intangible, financial or non-financial. Tangible or physical assets usually refer to equipment, inventory and properties owned by the organization. Physical assets are the opposite of intangible assets, which are non-physical assets such as leases, brands, digital assets, use rights, licenses, intellectual property rights, reputation or agreements. [26]

In the context of this research the assets are considered as physical items in factory environment. Physical industrial assets include various assets with different management decisions and data needs. To include the special features of these various assets we have divided physical assets into the following five categories: machinery and equipment, buildings, vehicles, inventories, and spare parts. As the research is limited to explore physical industrial assets in factory environment, the category of vehicles includes all mobile vehicles at the factory area, such as motor vehicles and trucks. Vehicles can be considered as assets as long they are transporting the property of the company to other destination, although the vehicle leaves the factory area. Therefore, the company's own railway wagons and trucks can be counted among assets in this research.

The asset group of inventories comprises four types of inventories: 1) finished goods, 2) work-in-process, 3) raw materials, and 4) maintenance and operating items, such as spare parts. Spare parts have been separated from inventories to be an own category. Spare parts are replacement items that are required to keep assets operating in a plant and they prevent excessive down-time in case of a breakdown [27]. The category of buildings at factory area includes factory buildings and other buildings, such as warehouses and office buildings. Machinery and equipment are the physical assets that are required in factory-specific processes. In addition to process-related machinery, this category includes also other equipment such as tools and computers.

Asset management is defined to be broader perception than the term of maintenance gives to understand. Asset management is considered as a set of activities associated with 1) identifying what assets are needed, 2) identifying funding requirements, 3) acquiring assets, 4) providing logistic and maintenance support systems for assets, and 5) disposing or renewing assets. Therefore, assets management aims to manage assets optimally and sustainably over the whole life cycle of assets. [28]

2.3 Matrix framework

In the context of this research, a matrix framework (Table 1) that combines the aspects of IoT technologies and asset management has been generated. A detailed literature review has been conducted to identify IoT based technologies and if they have been applied to the different asset groups and asset management. These findings have been included into the matrix framework (Table 1). Academic articles, in which IoT technology has been researched or applied to a specific asset group, are referred to in the matrix. The category of spare parts has been left outside the literature matrix, and is thus included in inventory category, due to the limited amount of research focusing on applying IoT technologies to spare parts.

According to the literature matrix, it can be stated that IoT technologies have been applied to a range of asset groups. RFID and WSN technologies have been researched widely and several applications have been identified. WSAN applications have not been studied widely but recently this technology has got more attention in literature. WSAN technology has not been applied to inventories or this just does not appear in literature. WPAN communication technologies have been applied in different contexts and especially the potential of ZigBee has been acknowledged in literature. Also NFC technology has been studied in literature and there are several applications. Barcodes have been in use for many years but 2D barcodes and their ability to store data and the potential of IP address to identify a certain object have become useful in the management of different asset groups. Localization technologies have also been applied to different asset groups and particularly in smart factory context the indoor localization applications stand out in literature. When examining asset groups, it can be noticed that machinery and equipment, buildings and inventories have most applications while vehicles and earlier mentioned spare parts do not have as many applications or they just not appear in academic publications.

Table 1. Literature matrix: IoT technologies applied to the management of asset groups.

IOT TECHNOLOGIES	ASSETS: Machinery and equipment	Buildings	Vehicles	Inventories
RFID	Remote condition monitoring, failure follow-up notifications, embedded health history with the asset [29, device management, equipment monitoring [21], maintenance information sharing platform [30], collecting real-time production information [31]	Access control system [32], [33]	Electric vehicle batteries [34]	Storage levels of parts, real-time information about products on assembly line [35], inventory control [36], [37], resource management system [38]
WSN	Condition-based maintenance [39],[11], fault diagnostics [40], collecting running parameters [41]	Energy monitoring, behavioral monitoring, space monitoring [20], energy management, power consumption monitoring [16]	Traffic estimation, traffic control [15]	Online inventory management system [42], inventory management [43]
WSAN	Gas detection and immediate isolation of gas leak source [44], process control applications [45]	Energy saving, maximization of the comfort in the building [20], temperature control in a work environment [19], power management [46], building automation [18]	Unmanned ground vehicle [47]	
WPAN (Bluetooth, Zigbee)	Collecting running parameters [41]	Bluetooth, Zigbee, Wifi: communication of smart living space [48]	Bluetooth-enabled headset and voice-activated features [49]	Inventory tracking with Zigbee [50]
NFC	Context-aware mobile support system [50], recurring maintenance processes: central process control and documentation [52]	Classroom access control [53] access keys for offices and houses [22]		Availability and stock information of products [54], inventory control [55]
Naming technologies, (barcodes, EPC, IP address)	Maintenance information sharing platform [30]	IP address: building automation [18]	IP address: tracking of individual vehicles [23]	2D barcodes: product information, mobile product verification [56]
Location based technologies (satellite, mobile networks, local area networks)	Tool tracking and localization with RF signals [57]	NFC smartphone indoor interactive navigation system [58]	Asset localization [24], GPS: location of vehicles [23]	Indoor locating with RF signals [36]

Based on the matrix framework, it can be concluded that 1) RFID technology has been widely researched and applied 2) WSN technology appears to be easily varied in different contexts, 3) WSAN has interesting applications and there might be untapped potential as automation can be employed more widely in the management of different asset groups e.g. in machinery and equipment, where automation can be used to prevent failures, or in vehicles, 4) NFC technology might have potential to be applied more widely in different contexts, such as in simplifying documentation during maintenance tasks and access control in various contexts, and 5) there are various applications for communication, naming, and localization technologies.

3 Empirical results framework

Empirical data have been gathered via an industrial expert panel that is comprised of industrial asset management and industrial maintenance specialists from five companies, representing original equipment manufacturers and customer companies from mining and energy industries. Mining and energy industries are traditionally asset and capital intensive industries where IoT technologies and automation create significant potential. In total, six experts from these companies participated in the panel. Experts were asked to evaluate the potential of IoT technologies in the management of different asset groups. The scale was 0–5, where 0 is "cannot say", 1 "no potential", 2 "some potential", 3 "potential", 4 "quite high potential", and 5 "high potential". The matrix (Table 2) sums up the views of experts, as an average number given by six experts. In addition to numbers, the potential is also illustrated by the shades of grey, darker grey representing higher potential and lighter grey representing less potential. White areas mean that potential is unclear or a large proportion of respondents, i.e. over half of the respondents could not say if there is potential, and therefore the average value could be distorted. The category of spare parts has been included here as a separate category because of the special attention given by experts.

Results indicate that 1) RFID technology is seen as a potential technology to be applied to different asset groups, 2) there is potential to utilize WSN in the management of machinery and equipment, buildings, and vehicles, 3) WSAN technology has potential to be applied especially to management of buildings, but to machinery and equipment and vehicles as well, 4) potential of WPAN has also been recognized, 5) NFC technology could be potential in the management of machinery and equipment, 5) naming technologies might be useful to important spare parts, inventories, and machinery and equipment, 7) localization technologies have been identified as potential technologies especially when tracking vehicles.

In addition to the summary matrix, experts have provided comments and usage examples of IoT technologies. For example, RFID technology could be applied to the control of raw material and spare part inventories. In addition, "RFID technology could benefit CBM calculation when operating hours can be targeted at each component". Regarding NFC technology, it has been said that "NFC can be utilized to access control and signing for working orders". NFC applications have been said to be "handy in mobiles and tablets, and therefore other separate devices are not needed". "NFC could work better in some contexts where RFID applications are inflexible to use." Regarding naming technologies, it has been said as follows: "while doing maintenance work, the equipment could be identified and then enter the information and conducted operations into the follow-up system". It was also said that "the most important spare parts should be named and this allows monitoring the spare part consumption of whole installed base" and "2D barcode could enable the access to the documentation". Even though the potential of localization technologies in the management of machinery and equipment is not high, it has been said that "sometimes it would be practical if maintenance worker could localize accurately the equipment that needs to be overhauled". However, apart from what mentioned above, the threats of IoT technologies have also been recognized: "If it is possible to control important assets, such as machinery, via IP address, information security challenges must be acknowledged".

Table 2. Potential of IoT technologies in the management of asset groups.

	ASSETS				
IOT TECHNOLOGIES	Machinery and equipment	Buildings	Vehicles	Inventories	Spare parts
RFID	3,0	3,7	4,0	3,7	3,0
WSN	4,4	4,0	4,7	2,7	2,0
WSAN	3,8	4,8	3,3	2,7	1,5
WPAN	4,8	4,0	4,0	2,7	2,8
NFC	3,5	2,3	3,5	2,5	3,0
Naming technologies	3,2	2,5	3,0	3,7	4,5
Localization technologies	2,8	2,7	4,2	4,5	2,0

4 Discussion and conclusions

IoT technologies and applications have been studied in detail. But this kind of summing-up literature review of IoT technologies and applications in asset management does not appear in the literature. Results establish understanding of how these technologies can be applied to the asset management of future smart factories and what technologies can be utilized to rationalize data acquisition and transmission in industrial applications. The research acts as a kick for interest to apply IoT technologies more in industry. The research also helps to identify the potential of IoT technologies for data processes and recognizes the themes of further research. Literature matrix and empirical matrix represent the responds to the both research questions. According to literature review, IoT technologies can be

applied widely in the management of different asset groups. Also industrial experts see potential to exploit technologies in their business environments.

If the literature matrix and expert views are compared, the main difference is related to spare parts. While research is lacking, industrial experts see potential for a number of technologies, such as naming technologies. A reason for this might be the fact that researches have not been focusing on spare part inventories while asset management practitioners are more interested in spare parts in particular. Another difference is that, based upon the literature, there are fewer IoT applications for vehicles than the views of experts indicate. This could be because the research is only just increasing or IoT innovations related to vehicles might be carefully protected by companies from publicity. Thus, there could be a lack of academic publications compared to e.g. machine or inventory-related applications.

IoT technologies have a huge potential but more needs to be done before applications can be exploited more widely in real industrial environments. An important issue is how collected data can be effectively turned into business information. This sets high level requirements for middleware layer which filters and processes collected data. Data is required to be accurate, adequate, and timely. Additionally, data need to be integrated as much as possible into same system or database in order to enable automation and ease of use.

The limitation of this research is the small size of industrial expert panel, and therefore more comprehensive empiric research need to be conducted. Further research could concern more comprehensive empiric research about the potential to apply IoT technologies in different business environments. The future research could also focus on examining the lack of research in the field of spare parts. It should be investigated more carefully if IoT applications for spare parts are not researched at all or if there is any potential to apply IoT technologies. The next step is also to study how all the collected data could be used effectively in applications and decision making.

5 References

[1] H. Kivimäki, T. Sinkkonen, S. Marttonen, and T. Kärri, "Creating a life-cycle model for industrial maintenance networks," in *Proc. 3rd Int. Conf. on Maintenance Performance Measurement and Management*, Lappeenranta, 2013, pp. 178–191.

[2] T. Sinkkonen, A. Ylä-kujala, S. Marttonen, and T. Kärri, "Better maintenance decision making in business networks with a LCC model," in *Proc. 4th Int. Conf. on Maintenance Performance Measurement and Management*, Coimbra, 2014, pp. 57–64.

[3] S. Marttonen, S. Monto, and T. Kärri, "Profitable working capital management in industrial maintenance companies," *J. of Quality in Maintenance Engineering*, vol. 19, no. 4, pp. 429–446, 2013.

[4] O. Vermesan and P. Friess, *Internet of Things – Converging Technologies for Smart environments and Integrated Ecosystems*, River Publishers, Aalborg. 2013

[5] M. M. Wang, J-N. Cao, J. Li, and S.K. Dasi, "Middleware for wireless sensor networks: a survey," *Journal of Computer Science and Technology*, vol. 23, no. 3, pp. 305–326, 2008.

[6] M. E. Porter and J. E. Heppelmann, "How smart, connected products are transforming competition," *Harward Business Review*, vol. 11, pp. 1–23, 2014.

[7] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: a survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[8] D. Miorandi, S. Sicari, F. De Pellegrini, and I. Chlamtac, "Internet of things: Vision applications and research challenges," *Ad Hoc Networks*, vol. 10, no. 7, pp. 1497–1516, 2012.

[9] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things: vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[10] S. Li, L. D. Xu, and S. Zhao, "The internet of things: a survey," *Information Systems Frontiers*, April 2014.

[11] E. Jantunen, C. Emmanouilidis, A. Arnaiz, and E. Gilabert, "Economical and technological prospects for e-maintenance," *International Journal of System Assurance Engineering and Management*, vol. 1, no. 3, pp. 201–209, 2010.

[12] H. Kopetz, *Real-time systems series: design principles for distributed embedded applications,* 2nd edition, Springer, New York, 2011.

[13] X. Zhu, S. K. Mukhopadhyay, and H. Kurata, "A review of RFID technology and its managerial applications in different industries," *Journal of Engineering and Technology Management*, vol. 29, no. 1, pp. 152–167, 2012.

[14] I. F. Akyildiz, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, pp. 393–422, 2002.

[15] M. Tubaishat, P. Zhuang, Q. Qi, and Y. Shang, "Wireless sensor networks in intelligent transportation systems," *Wireless Communications and Mobile Computing*, vol. 9, pp. 287–302, 2009.

[16] C. Jacquemod, B. Nicolle, and G. Jacquemod, "WSN for smart building application," in *Proc. 10th European Workshop on Microelectronics Education*, Tallinn, 2014, pp. 102–105.

[17] I. F. Akyildiz and I. H. Kasimoglu, "Wireless sensor and actor networks: research challenges," *Ad Hoc Networks*, vol. 2, pp. 351–367, 2004.

[18] M. Jung, C. Reinisch, W. Kastner, "Integrating building automation systems and IPv6 in the Internet of Things," in *Proc. 6th Int. Conf. on Innovative Mobile and Internet Services in Ubiquitous Computing*, Palermo, 2012, pp. 683–688.

[19] A. De Paola, S. Gaglio, G. Lo Re, and M. Ortolani, "Sensor9k: A testbed for designing and experimenting with WSN-based ambient intelligence applications," *Pervasive and Mobile Computing*, vol. 8, no. 3, pp. 448–466, 2012.

[20] G. Fortino, A. Guerrieri, G. M. P. O'Hare, and A. Ruzzelli, "A flexible building management framework based on wireless sensor and actuator networks," *Journal of Network and Computer Applications,* vol. 35, pp. 1934–1952, 2012.

[21] N. Wang, P. Guan, H. Du, and Y. Zhao, "Implementation of asset management system based on wireless sensor technology," *Sensors and Transducers*, vol. 164, no. 2, pp. 136–144, 2014.

[22] V. Conskun, B. Ozdenizci, and K. Ok, "A survey on near field communication (NFC) technology," *Wireless Personal Communications,* vol. 71, no. 3, pp. 2259–2294, 2013.

[23] R. I. Rajkumar, P. E. Sankaranarayanan, and G. Sundari, "GPS and Ethernet based real time train tracking system," in *Proc. Int. Conf. on Advanced Electronic Systems*, Pilani, 2013, pp. 282–286.

[24] A. Motamedi, M. M. Soltani, and A. Hammad, "Localization of RFID-equipped assets during the operation phase of facilities," *Advanced Engineering Informatics*, vol. 27, no. 4, pp. 566–579, 2013

[25] M. Chen, S. Mao, and Y. Liu, "Big data: a survey," *Mobile Networks and Applications*, vol. 19, no. 2, pp. 171–209, 2014.

[26] International Standard (ISO) 55000 (E), *Asset management – Overview, principles and terminology*, International Organization for Standardization, 2014.

[27] R. Gulati, *Maintenance and reliability best practices*, Industrial Press, New York, 2009.

[28] N. A. J. Hastings, *Physical Asset Management*, Springer London Dordrecht Heidelberg, New York, 2010.
[29] A. Haider and A. Koronios, "Potential uses of RFID technology in asset management," *Engineering Asset Management Review*, vol. 1, pp. 173–194, 2010.
[30] Y-C. Lin, W-F. Cheung, and F-C. Siao, "Developing mobile 2D barcode/RFID-based maintenance management system," *Automation in Construction*, vol. 37, pp. 110–121, 2014.
[31] M. Liu, Q. Wang, L. Ling, and M. Zhang, "RFID-enabled real-time manufacturing operation management system for the assembly process of mechanical products," *International Journal of RF Technologies: Research and Applications,* vol. 6, no. 4, pp. 185–205, 2015.
[32] Y. Qiu, J. Chen, and Q. Zhu, "Campus access control system based on RFID," in *Proc. IEEE 3rd Int. Conf. on Software Engineering and Service Science*, Beijing, 2012, pp. 407–410.
[33] Q. Zhu, H. Zhao, X. Shi, and R. Hu, "The UML model for access management system of intelligent warehouse based on RFID," in *Proc. Int. Conf. on Manufacturing Science and Technology*, Singapore, 2012, vols. 383-390, pp. 5855–5860.
[34] X. Wang, Q. Dang, J. Guo, and H. Ge, "RFID application of smart grid for asset management," *International Journal of Antennas and Propagation*, vol. 264671, pp. 1–6, 2013.
[35] S-C. Chen, C-Y. Chang, K-S. Liu, and C-W. Kao, "The prototype and application of RFID implementation: a case study of automobiles assembly industries," *International Journal of Electronic Business Management*, vol. 12, no. 2, pp. 145–156, 2014.
[36] C. C. Chang, P. C. Lou, and Y. G. Hsieh, "Indoor locating and inventory management based on RFID-Radar detecting data," *Journal of Applied Geodesy*, vol. 6, no. 1, pp. 47–54, 2012.
[37] C. M. Liu and L. S. Chen, "Applications of RFID technology for improving production efficiency in an integrated-circuit packaging house," *International Journal of Production Research*, vol. 47, no. 8, pp. 2203–2216, 2009.
[38] G. Liu, W. Yu, and Y. Liu, "Resource management with RFID technology in automatic warehouse system," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, 2006, pp. 3706–3711.
[39] A. Tiwari, F. L. Lewis, and S. G. Shuzhi, "Wireless sensor network for machine condition based maintenance," in *Proc. 8th Int. Conf. on Control, Automation, Robotics and Vision Kunming*, China, 2004.
[40] B. Lu and V. C. Gungor, "Online and remote motor energy monitoring and fault diagnostics using wireless sensor networks," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 11, pp. 4651–4659, 2009.
[41] C-L. Hsu, "Constructing transmitting interface of running parameters of small-scaled wind-power electricity generator with WSN modules," *Expert Systems with Applications*, vol. 37, pp. 3893–3909, 2010.
[42] S. Vellingiri, A. Ray, and M. Kande, "Wireless infrastructure for oil and gas inventory management," in *Proc. 39th Annual Conf. of the IEEE Industrial Electronics Society*, Vienna, pp. 5461–5466, 2013.
[43] A. Mason, A. Shaw, and A. I. Al-Shamma'a, "Inventory management in the packaged gas industry using wireless sensor networks," *Lecture Notes in Electrical Engineering,* vol. 64, pp. 75–100, 2010.
[44] A. Somov, A. Baranov, and D. Spirjakin, "A wireless sensor-actuator system for hazardous gases detection and control," *Sensors and Actuators A*, vol. 210, pp. 157–164, 2014.
[45] C. Lu, A. Saifullah, B. Li, H. Gonzalez, D. Gunatilaka, C. Wu, L. Nie, and Y. Chen, "Real-Time Wireless Sensor-Actuator Networks for Industrial Cyber-Physical Systems," *Proceedings of the IEEE*, to be published.
[46] N. K. Suryadevara, S. C. Mukhopadhyay, S.D.T. Kelly, and S.P.S. Gill, "WSN-based smart sensors and actuator for power management in intelligence buildings," *EEE/ASME Transactions on Mechatronics*, vol. 20, no. 2, pp. 564–571, 2014.
[47] J. Li and R. R. Selmic, "Implementation of Unmanned Ground Vehicle navigation in Wireless Sensor and Actuator Networks," in *Proc. 23rd Mediterranean Conference on Control and Automation*, Torremolinos2015, pp. 871–876.
[48] Z-Y. Bai and X-Y. Huang, "Design and implementation of a cyber physical system for building smart living spaces," *International Journal of Distributed Sensor Networks*, vol. 764186), pp. 1–9, 2012.
[49] S. M. Mahmud and S. Shanker, "In-vehicle secure wireless personal area network SWPAN," *IEEE Transactions on Vehicular Technology*, vol. 55, no. 3, pp. 1051–1061.
[50] H. Yang and S-H. Yang, "Connectionless indoor inventory tracking in Zigbee RFID sensor network," in *Proc. 35th Annual Conference of IEEE Industrial Electronics*, Porto, 2009, pp. 2618-2623.
[51] N. Papathanasiou, D. Karampatzakis, D. Koulouriotis, and C. Emmanouilidis, "Mobile personalised support in industrial environments: coupling learning with context – aware features," *IFIP Advances in Information and Communication Technology*, vol. 438, no. 1, pp. 298–306, 2014.
[52] S. Karpischek, F. Michahelles, A. Bereuter, and E. Fleisch, "A maintenance system based on near field communication," in *Proc. 3rd Int. Conf. on next generation mobile applications, services and technologies*, Cardiff, Wales, 2009, pp. 234–238.
[53] D. Palma, J. E. Agudo, H. Sánchez, and M. M. Macias, "An internet of things example: classrooms access control over near field communication," *Sensors*, vol. 14, no. 4, pp. 6998–7012, 2014.
[54] S. Karpischek, F. Michahelles, F. Resatsch, and E. Fleisch, "Mobile sales assistant: An NFC-based product information system for retailers," *in Proc. 1st International workshop on near field communication*, Hagenberg, 2009, pp. 20–23.
[55] R. Iqbal, A. Ahmad, and A. Gillani, "NFC based inventory control system for secure and efficient communication," *Computer Engineering and applications journal*, vol. 2, no. 1, pp. 23–33, 2014.
[56] J. Z. Gao, L. Prakash, and R. Jagatesan, "Understanding 2D-barcode technology and applications in M-commerce – Design and implementation of a 2D barcode processing solution," in *Proc. 31st Annual International Computer Software and Applications Conference*, Beijing, 2007, pp. 49–56.
[57] P. M. Goodrum, M. A. McLaren, and A. Durfee, "The application of active radio frequency identification technology for tool tracking on construction job sites," *Automation in Construction*, vol. 15, no. 3, pp. 292–302, 2006.
[58] J.H. Choo, S. N. Cheong, and Y.L. Lee, "Design and development of NFC smartphone indoor interactive navigation system," *World Applied Sciences Journal*, vol. 29, no. 6, pp. 738–742, 2014.

Efficient Algorithms for Mining DNA Sequences

Guojun Mao
Information Scholl, Central University of Finance and Economics, Beijing, P R China

Abstract - *Most data mining algorithms have been designed for business data such as marketing baskets, but they are less efficient for mining DNA sequences. Unlike transaction sequences in business, DNA sequences typically have a small alphabet but a much long size, and so mining DNA sequences faces different challenges from other applications. This paper deals with the problem of mining key segments from DNA sequences, and by designing a compact data structure called Association Matrix, our algorithms maintains a less memory consumption as well as get more precise mining results. The Association Matrix is a novel in-memory data structure, which can be proved by experiments so compact that it can deal with super long DNA sequences in a limited memory usage. Using sliding window techniques, we can also transfer a super long DNA sequence into a series of formal short sequences. Thus, we not only may process a single long DNA sequence in a high efficiency, but also can mine valuable patterns from multiple length-varied DNA sequences. Based on these models, we design the algorithms for mining key segments from a single long DNA sequence and from multiple DNA sequences, and related experiments show these algorithms are scalable with changing of different minimum association degrees.*

Keywords: Data mining, mining DNA sequence, association matrix, key segment.

1 Introduction

The development of molecular biology in the last decades has made various biological data coming. Using data mining techniques to analyze biological data has been becoming an important research problem. However, it is fact that most classical data mining algorithms were designed for business transaction data as the first motivation, and it also has been proved that they are not more efficient for analyzing DNA sequences.

In general, a DNA sequence can be represented as an alphabetical string in the biological databases, but such a string often has different features from a business transaction sequence.

First, the DNA sequences always have a small alphabet. That is, there are four different nucleic acids: Adenine(A), Thymine(T), Guanine(G), and Cytosine(C). In contrast, in a marketing database, the transaction sequences are defined in a large alphabet that represents hundreds or thousands goods.

Second, a DNA sequence often is a very long sequence. For example, the human genome is made of roughly three billion of nucleic acids. In contrast, in a marketing database, a transaction sequence mostly is comprised of a lot of shorter sequences from 10 to 20 items.

Third, a super long DNA sequence with a small alphabet often contains a few of appearance-frequent short sequences (Segments), which always pay important structural or functional roles. Therefore, finding such segment patterns from the DNA sequences should be a different type of study project with other business data mining.

According to the special features of DNA sequences, this paper will design a called Association Matrix data structure, and making use of such a data structure, some novel algorithms to efficiently mining key segments from one or multiple DNA sequences will be constructed.

The contributions of this paper are as follows:

- It introduces a novel data structure called Association Matrix for mining key segments from DNA sequences. To the best of our knowledge, this is the first such structure to mining DNA sequences, and it also has many potential applications in analyzing other super long sequences.
- Using the sliding windows technique, multiple length-varied DNA sequences can be transferred into a series of formal short sequences, and so the efficient and effective algorithm for mining common patterns from multiple DNA sequences is constructed.

The rest of this paper is organized as follows. Section 2 gives the related work introduction. In section 3, we present basic theoretic methods for abstracting and expressing related problems. Section 4 gives the algorithms for mining key segments from a long DNA sequence or from multiple length-varied DNA sequences. In Section 5, we evaluate the performance of the proposed methods by experiments. Section 6 concludes this paper and future work.

2 Related Work

In bioinformatics, finding similarity of several sequences has been broadly studied, and a detailed survey on this technique was given by Hirosawa [1]. When an entire sequence is similar to another, their similarity is useful, but their local relations are difficult to be found by similarity computing. In fact, many biological problems such as poly-regions in DNA need search out important segments on a

DNA sequence. Papapetrou et al developed three efficient detection methods for it [2]: The first applies recursive segmentation that is entropy-based; the second uses a set of sliding windows that summarize each sequence segment using several statistics; the third employs a technique based on majority vote. These methods provide the basic ideas of mining key segments from DNA sequences.

Applying data mining techniques to DNA sequences has also become an important research focus. Specially, with appearance and development of DNA sequences or genomic databases, data mining can provide a basic technical support in view of computing bio-information. For example, Bell et al used frequency mining methods to discover common strings from DNA sequences [3].

In fact, mining frequent sub-sequences and association rules is a basic and important task in data mining, and it is a distillation of such techniques to detect strings that are very repetitive within a given sequence or across sequences. In addition, the problem of principal component analysis and discriminated analysis of DNA features were presented, some of which employed sequence classification techniques of data mining [4], [5]. Habib et al gave the methods of DNA motif comparison that was based on Bayesian algorithms of data mining [6].

From the view of data miming, one related problem to this paper is to mining frequent sequences from sequential databases. The concept of sequential patterns was first introduced and discussed as a data mining problem in 1995 [7]. Algorithm GSP was developed for mining sequential patterns that is a breadth-first search and button-up method [8]. Free-Span is another efficient algorithm for mining sequential patterns [9], which has less effort than GSP in candidate sequence generation, but still makes use of the spirit of GSP. Up to now, many effective algorithms for mining sequential data were presented [10]-[13]. The above research jobs were mainly based on business transaction databases, and focused on improving the performance of mining sequential databases that often include many shorter sequences.

Another relation-closed technique with this paper is mining frequent sub-sequences from long sequences. A series of research efforts on this field has been made by Mannila and his colleagues, including Bayesian analysis techniques on event sequences [14], frequent episodes in event sequences [15], and similarity evaluation between event sequences [16]. Other typical works include: segmenting long time series in an online way [17]; mining common rules from multiple sequences with the window size constraint 18]; finding frequent sequential patterns over a large scale of data by using Path-Tree [19]; discovering frequent patterns with periodic wildcard gaps [20].

3 Terminology and definitions

As is known to all that a cell uses DNA to store their genetic information, and a DNA molecule is composed of two linear strands coiled in a double helix. Each strand is made of a linear sequence with adenine(A), thymine(T), cytosine(C), or guanine(G), and two strands abide by base pairing rules (A with T and C with G). Therefore, modern bioinformatics has organized a DNA molecule into a character string and stored them in databases in order to be used in science research.

Definition 1 (DNA Sequence). Given alphabet set {A , G, C, T}, a DNA sequence is denoted by $s = <x_1, x_2, \ldots, x_L>$, $x_i \in$ {A, G, C, T} for all i =1, 2, … , L. Also, for any a sequence $t= <y_1, \ldots, y_{k-1}, y_k>$ in {A, G, C, T}, if exists i in s to have $x_{i+j-1} = y_j$ (j=1, 2, .., k), then t is called a sub-sequence of s, and $< y_1, \ldots, y_{k-1}>$ is called the *Prefix* of t about s, represented as *Prefix*(t); y_k is called the *Postfix* of t, represented as *Postfix*(t); Thus, a sub-sequence t of a DNA sequence s can be through a postfix-connection operation ∞ to generate: $t = Prefix(t) \infty Postfix(t)$.

Definition 2 (Association Matrix). Given a DNA sequence $s = < x_1, x_2, \ldots, x_L>$, an *Association Matrix* for it is defined as ($p_{i,j}$), where: each row element is related to the length-fixing strings of {A , G, C, T}, and if the fixing length is k, it is called a k-level association matrix. Also, it always has 4 column element, related to Letter A, G, C or T; each matrix element $p_{i,j}$ is an Integer, which represents the appearing number of the sub-sequence $i \infty j$ in the DNA sequence.

As a novel and important data structure, the Association Matrix can provide an efficient information abstract from scanning the original long DNA sequence, and will further support pattern mining from the DNA sequences.

Example 1. Considering the DNA sequence s = <ATGTCGTGATTGCATTACTACT>, its 1-level association matrix is shown in Fig. 1.

	A	T	C	G
A	0	3	2	0
T	2	2	1	3
C	1	2	0	1
G	1	2	1	0

Fig. 1. The 1-level Association Matrix for the DNA sequence in Example 1.

Definition 3 (Key Segment). Given an association matrix ($p_{i,j}$) on a DNA sequence. Set a minimum association threshold be *Min-Ass,* when $p_{i,j} >= Min\text{-}Ass$, then $i \infty j$ is thought as a Key Segment of s.

Indeed, mining key segments from DNA sequences is our main target in this paper. By making use of the association matrix structure in Definition 2, it is easy to evaluate the occurring frequency of any sub-string in an original DNA sequence, and so key segments in a DNA sequence can be found out.

Example 2. For Fig. 1, if *Min-Ass* = 2, then its 2-length key segments can be obtained by scanning the 1-level association matrix: <AT>, <AC>, <TA>, <TT>, <TG>, <CT>, and <GT>.

Definition 4 (Maximum Key Segment). Given a DNA sequence *s*. A key segment is called a Maximum Key Segment only when it is not contained by any other key segments of *s*.

4 Algorithms

Large-size DNA samples can derive from a multitude of diverse organisms, and a key problem is to functionally search out key sub-sequences that is often much shorter but appearance-frequent. Such short sequences are called key segments in this paper. In this section, we first design the algorithm for discovering key segments from a single DNA sequence. Then, through analyzing the key problems of concurrently mining multiple DNA sequences, discuss the related mining methods.

4.1 Mining key segments from a DNA sequence

Naturally, an iteration procedure is necessary to find out key short sequences from a long sequence. As an instance, Example 2 has generated 2-length key segment set in the DNA sequence, so we can continue to do iterations to this dataset, in order to discover other key segments with longer sizes.

Example 3. For DNA sequence *s* in Example 1, we have gotten its 2-length key segment set: {<AT>, <AC>, <TA>, <TT>, <TG>, <CT>, <GT>}, so its 2-level association matrix can be further constructed shown as Fig. 2 (a). Going a step further, its 3-length key segment set is: {<ATT>, <ACT>, <TAC>}, and so its 3-level association matrix can be written as Fig. 2 (b). Scanning the 3-level association matrix, its one 4-length key segment <TACT> is found. Duo to its 4-level association matrix degenerates into a vector (1,0,0,0), so no 5-length key segment is found and the iteration is stopped.

	A	T	C	G
AT	0	2	0	1
AC	0	2	0	0
TA	0	0	2	0
TT	1	0	0	1
TG	1	1	1	0
CT	1	0	0	0
GT	0	0	1	1

(a)

	A	T	C	G
ATT	1	0	0	1
ACT	1	0	0	0
TAC	0	2	0	0

(b)

Fig. 2. The Association Matrixes of 2-level and 3-level for the DNA sequence in Example 1

Obviously, the Association Matrix structure is simple, but it can be efficient for finding key segments from a DNA sequence. Therefore, based on association matrix as an in-memory structure, we can organize related association information scanning from a long DNA sequence to the main memory, and so important segments can be efficiently searched out. Based on the step-by-step iteration idea, we can design the effective algorithm to mine a DNA sequence. Algorithm 1 provides the pseudo code for mining key segments from a single long DNA sequence.

Algorithm 1. Mining key segments from a DNA sequence.

```
INPUT: DNA sequence s; minimum association Min-Ass.
OUTPUT: s' key segments KS.
begin
k←1; m←4; row-set←{A,T,G,C};
WHEN row-set is not null DO
   generate the s' k-level Association Matrix: (p_{i,j})_{m*4};
  row-set←{};
   FOR i=1 TO m
     FOR j=1 TO 4
        IF pi,j >= Min-Ass THEN insert i∞j into row-set;
   add all elements of row-set into KS;
   updating m with the size of row-set;   k++;
ENDDO
Return KS.
end.
```

Example 4. For DNA sequence *s* in Example 1, applying Algorithm 1, all key segments can be obtained: {<AT>, <AC>, <TA>, <TT>, <TG>, <CT>, <GT>, <ATT>, <ACT>, <TAC>, <TACT>}; and its maximum key segment set is: {<TG>, <GT>, <ATT>, <TACT>}.

4.2 Mining key segments from multiple DNA sequences

To understand the common gene characters in multiple DNA sequences, it is necessary to together mining them. However, the job can be more difficult. This is because they can have different lengths as well as they can be very long. For solving this problem, we use sliding window technique, which can make multiple length-varied DNA sequences becoming more formal mining objects.

Based on sliding window, we can transfer a super long DNA sequence into a series of shorter sequences. Such short sequences are more formal and length-fixing, and so it is possible to process them in limited main memory more efficiently than to directly do these super long sequences.

Definition 5 (DNA Short Sequence). Given an original DNA long sequence $s = <x_1, x_2, \ldots, x_L>$ and the size of the sliding window *K*, if *L* is much larger than *K*, the set of short sequences for *s* can be built up by using sliding window technique. That is, *s* is transformed into the set of short sequences with the fixing length *K*: $\{s_i\}_{L-K+1}$, where each $s_i = < x_i, x_{i+1}, \ldots, x_{i+K-1}>$ is called the *i*th window sequence.

Definition 6 (Association Matrix of Short Sequence Set). Given the short sequence set $sSet=\{s^1, s^2, \ldots, s^n\}$. For each short sequence s^k (k=1, 2, ..., n), its association matrix $(p_{i,j})$ can be obtained by :

$$(\sum_{k=1}^{n} p_{i,j}^{k}) \quad (1)$$

Where $p_{i,j}^{k}$ is the appearing number of the string $i \infty j$ in s^k (described as the above Definition 2).

Example 5. Set the size of the sliding window be 10. Supposed the following three DNA sequences:

(1) <ATGTCGATTGCAAGCTGCG>;
(2) <AGTCGATGCATGATCG> ;
(3) <CGTCACTGATATG>.

Then, using sliding window technique, we can get the set of short sequences in every window from these three long sequences:

(1) The 1st window: {<ATGTCGATTG>, <AGTCGATGCA>, <CGTCACTGAT>};

(2) The 2nd window: <TGTCGATTGC>, <GTCGATGCAT>, <GTCACTGATA>};

(3) The 3rd window: {<GTCGATTGCA>, <TCGATGCATG>, <TCACTGATAT>};

(4) The 4th window: {<TCGATTGCAA>, <CGATGCATGA>, <CACTGATATG>};

(5) The 5th window: {<CGATTGCAAG>, <GATGCATGAT>, <ACTGATATGX>};

(6) The 6th window: {<GATTGCAAGC>, <ATGCATGATC>, <CTGATATGXX>};

(7) The 7th window: {<ATTGCAAGCT>, <TGCATGATCG>, <TGATATGXXX>};

(8) The 8th window: {<TTGCAAGCTG>, <GCATGATCGX>, <GATATGXXXX>};

(9) The 9th window: {<TGCAAGCTGC>, <CATGATCGXX>, <ATATGXXXXX>};

(10) The 10th window: {<GCAAGCTGCG>, <ATGATCGXXX>, <TGXXXXXX>}.

Note that: there is Letter X excepting {A, T, C, G} in the above window data, which is not any meaning but just fills the vacancy positions.

For a fixed window, the investigated data can organized into a set of short sequences from multiple DNA sequences by Definition 5. Also, when scanning the short sequence set in a window, its association matrix can be built according to Definition 6. Thus, its key segments in this window can be found out according to the idea of Algorithm 1.

For example, suppose *Min-Ass* =3, as far as the first window data in Example 5 is concerned, we can deal with it in a loop way with increasing the sizes of strings in the window. Fig. 3 shows related processing detail. Through iteratively computing in the first window, the key segment set in this window is found: {<AT>,<TC>,<TG>, <CG>, <GA>, <GT>, <GAT>,<GTC>}; and the maximum key segment set is {<TG>, <CG>, <GAT>,<GTC>}.

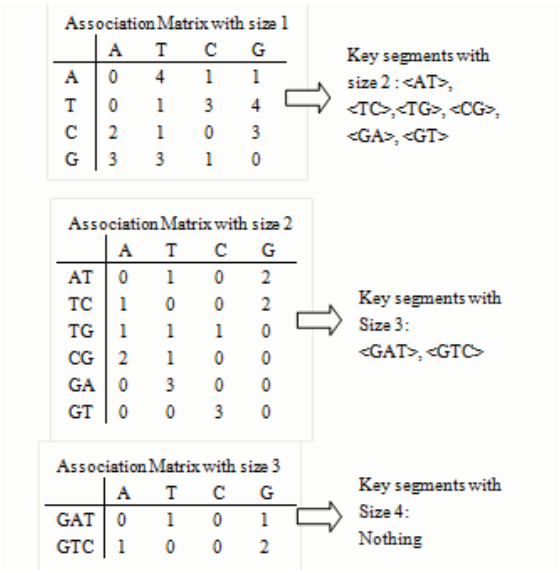


Fig. 3. The process for getting key segments from a window of multiple sequences (for Example 5, the first window, *Min-Ass* =3).

Algorithm 2 gives the pseudo code for mining key segments from multiple long DNA sequences. Algorithm 2 will deal with each window data in a loop way. And for each window, it need do three main things: (1) generating window data of short sequences; (2) finding key segments from the window data, whose processing phases include making and using the association matrix to store and search, so its basic ideal has been introduced in the above Algorithm 1; (3) Uniting the found key segments in all windows to get global key segments.

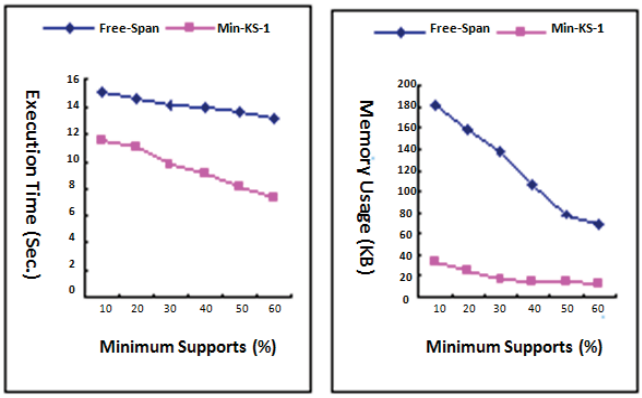
Algorithm 2. Mining Key Segments from Multiple DNA Sequences.

INPUT: DNA sequences s^1, s^2, s^3, …, s^n; Sliding window size K;
Minimum association *Min-Ass*.
OUTPUT: Key segments *KS*.
begin
k=1;
REPEAT
clear *sSet*;
generate the short sequence set of the kth window *sSet*;
generate the key segment set KS_k from *sSet* by Definition 6 and *Min-Ass*;
insert KS_k into *KS*;
k++; move to the next window of the DNA sequences
UNTIL all data is processed
Return *KS*;
end.

5 Evaluations of experiments

To evaluate the proposed algorithms, the above Algorithm 1 and 2, denoted as Min-KS-1 and Min-KS-*n*. We compared these algorithms with the popular sequential pattern mining algorithm Free-Span [9], and conducted several experiments on an 800MHz CPU with 2GB main memory.

Experiment 1. The first set of experiments was conducted on the synthetic data set: C256S64N4D100K, which have average length 256, whole volume 100K, and there are 4 sequences to simulate biology objects [21]. We conduct Min-KS-1 and Free-Span on the 4 sequences with different minimum Association-degrees (or Support-degree stated as Free-Span). Fig. 4 (a) and (b) show the results of execution time and memory space consumptions. Note that when setting a minimum association or support degree, Min-KS-1 and Free-Span are all executed on the 4 sequences, and the time consumption in Fig. 4 (a) is the average of time-spends on the 4 sequences, but the memory consumption is the maximum spending on the 4 sequences.

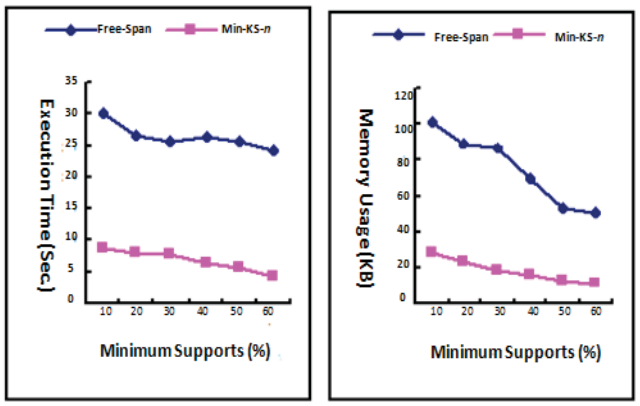


(a) (b)

Fig. 4. Time and space consumptions of Min-KS-1, and Free-Span on dataset C256S64N4D100K.

Result analysis of Experiment 1: From Fig. 4 (a), with increasing minimum Association degrees, both Min-KS-1 and Free-Span can make time down, but Min-KS-1 has much less time consumption than Free-Span on every minimum support degrees. Fig. 4 (b) shows that in comparison with Free-Span, main memory consumptions of Min-KS-1 are much less on every minimum degrees and more stable with varying minimum support degrees. A main factor contributed to these results is introducing Association matrix structure, which can be more compact than other data structure such that used in Free-Span.

Experiment 2. The second set of experiments was conducted on real life DNA sequences [21], which was organized into 125 DNA sequences (from size 2000 to 3000). We choose 5 sequences and use sliding window technique (Size 50) to evaluate the performance of Min-KS-*n*. Comparing algorithm is still Free-Span. Of course, Free-Span also has to be executed once for every window dada as well as Min-KS-*n* do. Fig. 5 (a) and (b) show the results of execution time and memory space consumptions of Min-KS-*n* and Free-Span, aiming to evaluate their scalability for mining key segments from multiple DNA sequences, with different minimum support degrees.



(a) (b)

Fig. 5. Execution time and memory space for Min-KS-*n* and Free-Span on real life DNA sequences.

Result analysis of Experiment 2: Fig. 5 shows, from left to right, with increasing minimum support degrees, executing time and using space consumptions of the two algorithms are declining, but our method are much better than Free-span.

6 Conclusions

DNA sequences are a different type of explosion of search space from classic transaction sequences, and so traditional data mining techniques for business transactional data are not more efficient for them. This paper has studied the challenges and methods for mining key segments from long DNA sequences.

We have presented a methodology to systematically mine DNA sequences. First, the structure Association Matrix aims at processing long sequences that have small letter set. Then, two main algorithms are presented to make this preliminary idea programmable. Algorithm 1 is to mine a single long DNA sequence, which introduces the basic procedure to discover key segments in a DNA sequence. By using the sliding window technique, Algorithm 2 implements finding key segments from multiple DNA sequences. In addition, experiments on both synthetic and real life data sets also demonstrated the proposed methods have less time costs and smaller space usages than some existing algorithms.

We are working on more experiments and graph layout algorithms in this research field. We will also investigate more biological sequences and do research to them.

7 Acknowledgment

First of all, we would like to extend my sincere gratitude to my colleagues including Dr. Wang and Dr. Liu, for her useful suggestions to this paper.

Second, high tribute shall be paid to two postgraduate students, Ms. J.Wang and Ms. S. Xie, for they help the author

to implement programming of the algorithms and do related experiments of this paper.

Finally, I am also deeply indebted to the NSFC (National Science Foundation of China), for this work was supported in part by the NSFC 61273293. Also, this work is partly supported by the CUFE discipline construction.

8 References

[1] M. Hirosawa, Y. Totoki, M. Hoshida and M. Ishikawa, "Comprehensive study on iterative algorithms of multiple sequence alignment)," J. Comput. Applic. Biosci., Vol.11, pp. 13-18, 1995.

[2] P. Papapetrou, G. Benson and G. Kollios, "Mining poly-regions in DNA," Intl J. Data Min Bioinform. Vol. 6, Issue 4, pp. 406-418, 2012.

[3] D. Bell and J. Guan, "Data mining for Motifs in DNA sequences," Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing: Lecture Notes in Computer Science, Vol. 2639, pp. 507-514, 2003.

[4] Z. Liu, D. Jiao and X. Sun, "Classifying genomic sequences by sequence feature analysis," Genomics Proteomics Bioinformatics, Vol. 3, Issue 4, pp. 201-205, 2005.

[5] P. Stegmaier, A. Kel, E. Wingenderand J. Borlak, "A discriminative approach for unsupervised clustering of DNA sequence motifs," PLoS Comput Bio, Vol. 9, Issue 3, e1002958-e1002958 , 2013.

[6] N. Habib, T. Kaplan, H, Margalit and N. Friedman, "A novel Bayesian DNA motif comparison method for clustering and retrieval," PLoS Comput Biol, Vol. 4, Issue 2, e1000010-e1000010, 2008.

[7] R. Agrawal and R. Srikant, "Mining sequential patterns," in 1995 Proc. Intl. Conf. on Data Engineering, IEEE Computer Society Press (Taipei, Taiwan), pp.3~14.

[8] R. Srikant and R. Agrawal, "Mining sequential patterns: generalizations and performance improvements," in 1996 Proc. Intl Conf. on Extending Database Technology (EDBT), Springer-Verlag Press, pp.3–17.

[9] J. Han and J. Pei, "Free-span: Frequent pattern-projected sequential pattern mining," in 2000 Proc. Intl. Conf. Knowledge Discovery and Data Mining, ACM Press (New York) , pp.355-359.

[10] J. Mohammed, "SPADE: an efficient algorithm for mining frequent sequences," J. Machine Learning, Vol. 42, Issue 1, pp. 31-60, 2001.

[11] J. Pei, J. Han, B. Mortazavi-Asl, J. Wang, H. Pinto, Q. Chen, U. Dayal and M. Hsu, "Mining sequential patterns by pattern-growth: the PrefixSpan approach," IEEE Transactions on Knowledge and Data Engineering, Vol. 16, Issue 11, pp. 1241-1440, 2004.

[12] W. Peng and Z. Liao, "Mining sequential patterns across multiple sequence databases," Data & Knowledge Engineering, Vol. 68, Issue 10, pp. 1014–1033 , 2009.

[13] C. Liu, L. Chen, Z. Liu and V. Tseng, "Effective peak alignment for mass spectrometry data analysis using two-phase clustering approach," Intl J. Data Mining and Bioinformatics, Vol. 9, Issue 1, pp. 52-66, 2014.

[14] E.Arjas, H.Mannila, M.Salmenkivi, R. Suramo and H. Toivonen. BASS: Bayesian analyzer of event sequences, in Proc. of COMPSTAT'96, A. Prat (ed.) (Barcelona, Spain, 1996), pp.199-204.

[15] H. Mannila, H. Toivonen and I. Verkamo, "Discovery of frequent episodes in event sequences," J. Data Mining and Knowledge Discovery, Vol. 1, Issue 3, pp. 259-289, 1997.

[16] H. Mannila. and M. Salmenkivi, "Finding simple intensity descriptions from event sequence data," in 2000 Proc. ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, ACM Press, 2000 (New York), pp.341-346.

[17] E. Keogh, S. Chu, D. Hart and M. Pazzani, "An online algorithm for segmenting time series," in 2001 Proc. IEEE Intl Conf. on Data Mining, IEEE Computer Society Press (San Jose, California, USA,), pp.289-296.

[18] P. Fournier-Viger, X. Wu, V. Tsen and R. Nkambou, "Mining sequential rules common to several sequences with the window size constraint," Lecture Notes in Computer Science, vol. 7310, pp. 299-304, 2012.

[19] G. Lee, Y. Chen and K. Hung, "FPTree: Mining sequential patterns efficiently in multiple data streams environment," J. Information Science and Engineering, Vol. 29, Issue 6, pp. 1151-1169, 2013.

[20] Y. Wu, L. Wang, J. Ren, W. Ding and X. Wu, "Mining sequential patterns with periodic wildcard gaps," Appl. Intell. , Vol. 41, Issue 1, pp. 99-116, 2014.

[21] K. Wang, Y. Xu and J. Yu, "Scalable sequential pattern mining for biological sequences," in 2004 Conf. t Intl Conf. on Information and Knowledge Management, ACM Press (Washington, DC, USA), pp.178-187.

Approximation Algorithms for D-hop Dominating Set Problem

Alina Campan, Traian Marius Truta, and Matthew Beckerich

Abstract—**Social networks are increasingly used today and their influence competes with well-established media outlets such as television and radio. The problem that we address in this paper is determining efficiently a good approximation of the network minimum *d*-hop dominating set. For this, we extend three existing dominating set algorithms to the context of d-hop dominating set problem and we perform a series of experiments on both real and synthetic data using these algorithms. For algorithms extension, we generate the d-closure of the network before applying the existing algorithms to determine good approximations to the *d*-hop dominating set problem. For experiments, we use several real networks datasets made available by the Stanford Network Analysis Project and we also generate synthetic networks using both random and power-law models. Our experiments show that the selected algorithms can efficiently find quality approximations for the minimum-size *d*-hop dominating set problem. The best algorithm choice is dependent on the network characteristics and the order of importance between the size of the *d*-hop dominating set and the time required to determine such a set.**

I. Introduction

SOCIAL networks are increasingly used today and their influence competes with well-established media outlets such as television and radio. For instance, social networks are used extensively for political campaigns such as United States 2016 republican and democratic primaries [20]. Even in authoritarian regimes social networks have an increasing influence over the population and they are used to increase the political awareness of users in such countries [18]. As a result, social networks are extensively studied in order to find methods to maximize such influences.

One problem that was extensively studied in the literature is determining efficiently the network minimum dominating set. We formalize this problem as follows. For $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ is the set of nodes and $\mathcal{E}$ is the set of edges, a set of nodes $\mathcal{DS} \subseteq \mathcal{N}$ is called a ***dominating set*** in $\mathcal{G}$ if every node $x \in \mathcal{N}$ is either in $\mathcal{DS}$ or is adjacent to a node in $\mathcal{DS}$ [3]. A dominating set with the smallest size is called ***minimum dominating set*** ($\mathcal{MDS}$). This problem has been shown to be an NP-complete problem [10], and various approximation algorithms have been proposed in order to obtain dominating sets close in size to the minimum dominating set [6, 9, 14, 16, 17].

In some networks dominating sets tend to be large and it will be enough if every node is not by the immediate neighbor but by node that lies at a distance of at most d from the original node. This modified domination problem was already introduced in the literature in the context of wireless networks [1]. Formally, a set $\mathcal{HDS}$ of nodes from $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ is called a ***d-hop dominating set*** if every node $x \in \mathcal{N}$ is either in $\mathcal{HDS}$ or there is a path between x and a node in $\mathcal{HDS}$ of size at most d. A d-hop dominating set with the smallest size is called ***minimum d-hop dominating set*** ($\mathcal{MHDS}$). Similarly to the original dominating set problem, determining $\mathcal{MHDS}$ is an NP-complete problem [1].

While the general minimum dominating set has received a lot of attention, there are fewer researchers that addressed the d-hop dominating set problems. As already mentioned before, this problem was introduced in [1] in order to determine the clusterheads in a wireless network. A local heuristic algorithm has been introduced to determine such clusterheads based on local cooperation between wireless nodes [1]. Newer distributed algorithms for producing a d-hop dominating set in a wireless network were analyzed in the wireless networks community [13, 19]. Outside wireless networking field, this problem is also analyzed from a specific graph model (directed graph with indegree bounded by one) in [2]. In the social networking context the closest work we are aware of is the problem of viral marketing where a d-hop dominating set can be seen as the subset of vertices that can activate all vertices in the network within at most d rounds [8]. Variants of d-hop dominating set problem also exists such as d-hop k-dominating set, connected d-hop dominating set, and connected d-hop k-dominating set [21].

Our contributions to this paper are to extend three dominating set algorithms from [6] to the context of d-hop dominating set problem and to perform a series of experiments on both real and synthetic data. For the first contribution, we use the d-closure of the social network [19] (see complete presentation in Section II) in order to apply the existing algorithms to determine good approximations to the dominating set problem. For experiments, we used several real networks datasets made available by the Stanford Network Analysis Project [10] and we also generated synthetic networks using both random and power-law models.

The remaining of this paper is structured as follows. Section II presents the three algorithms that are used in this paper. Section III describes the social networks used in our experiments the results of our experiments. Section IV presents conclusions and future work directions.

A. Campan (e-mail: campana1@nku.edu phone: +1-859-572-5776; fax: +1-859-572-5398,), T. M. Truta (e-mail: trutat1@nku.edu), and M. Beckerich (e-mail: beckerichm1@mymail.nku.edu) are with the Department of Computer Science, Northern Kentucky University, Nunn Drive, Highland Heights, KY 41099, USA.

II. D-HOP DOMINATING SETS ALGORITHMS

For a social network $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ is the set of nodes and $\mathcal{E}$ is the set of edges, we construct the network $\mathcal{G}_d = (\mathcal{N}, \mathcal{E}_d)$, where $\mathcal{E}_d = \{(x, y) \mid \delta(x, y) \leq d \text{ and } x \neq y\}$ and $\delta(x, y)$ represents the distance from the original network $\mathcal{G}$ between vertices x and y. The network $\mathcal{G}_d$ is called the d-closure of $\mathcal{G}$ [19]. We illustrate in Fig. 1 a network and its 2-closure. The newly added edges are represented by red dashed lines.

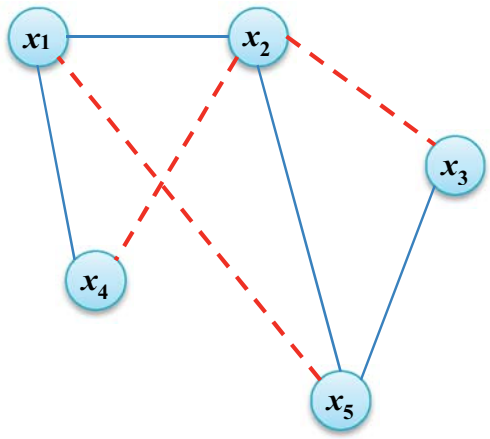


Fig. 1. A network $\mathcal{G}$ and its 2-closure, $\mathcal{G}_2$. The original edges from $\mathcal{G}$ are dark blue and the newly added edges in $\mathcal{G}_2$ are dashed red.

Based on the above construction, it is trivial to notice that any dominating set in $\mathcal{G}_d$ is a d-hop dominating set in $\mathcal{G}$, thus we can determine a d-hop dominating set in two easy steps. First, we construct the d-closure of the original network, and second we apply any existing dominating set algorithm to the newly created $\mathcal{G}_d$ network.

We use in this paper three algorithms for efficiently finding a close approximation to the minimum d-hop dominating set problem. These algorithms were presented in [6] for the general dominating set problem and we briefly describe them in the following paragraphs. We also include in these algorithms the creation of the d-closure network.

After the generation of the $\mathcal{G}_d$ network, the first algorithm (***Alg. 1***) selects in a for loop one node that covers the maximum number of currently uncovered nodes in the network. To determine if the nodes are in the dominating set, covered by a node in the dominating set, or not yet covered, we use a coloring scheme. Namely, nodes in the dominating set are colored as *black*, nodes covered by existing nodes in the dominating set are *gray*, and the remaining nodes are *white*. $\mathcal{W}(x)$ denotes the set of white nodes among the direct neighbors of x, including x itself.

```
Alg. 1 (G, HDS) is
  Input:  G = (N, E) - a social network;
          d - the d-hop parameter;
  Output: HDS - a d-hop dominating set for G;
  Create Gd = (N, Ed) - the d-closure of G.
  HDS = ∅;
  while □ white nodes in Gd do
    choose v □ {x □ N | w(x) = max u □ N |W(u)|};
    HDS:= HDS ∪ { v };
    For all neighbors x of v in Gd
      Color x as gray;
    End for;
    // Delete the vertex v and its adjacent edges.
    Gd = (N \{v}, Ed \{(x, y) □ Ed| x = v or y = v)});
  end while;
end Alg. 1.
```

In the second algorithm (***Alg. 2***) after the d-closure generation $\mathcal{G}_d$, we aim to improve the running time of ***Alg. 1*** by removing both the black and gray nodes from the remaining network. In this algorithm the use of node coloring is no longer useful. The degree function represents the number of adjacent nodes in the d-closure network. Its pseudocode is shown below:

```
Alg. 2 (G, HDS) is
  Input:  G = (N, E) - a social network;
          d - the d-hop parameter;
  Output: HDS - a d-hop dominating set for G;
  Create Gd = (N, Ed) - the d-closure of G.
  HDS = ∅;
  while N ≠ ∅ do
    choose v □ {x □ N |
              degree(x) = max u □ N (degree(u))};
    HDS:= HDS ∪ { v };
    // Delete v, Neighbors({v}), and their edges.
    // The remaining edges are labeled E'
    Gd = (N \{v}\ Neighbors({v}, E');
  end while;
end Alg. 2.
```

In the last algorithm (***Alg. 3***) after the d-closure generation $\mathcal{G}_d$, we consider the nodes in $\mathcal{N}$ in non-increasing order of the degree (again from the d-closure network) $v_1, v_2, \ldots, v_{|\mathcal{N}|}$, with $degree(v_1) \geq degree(v_2) \geq \ldots \geq degree(v_{|\mathcal{N}|})$. We pick the smallest i such that $|\cup_{j \leq i} Neighbors\ (v_j)| = |\mathcal{N}|$ and we take the subset $\{v_1, v_2, \ldots, v_i\}$ to be the d-hop dominating set of the graph. From this set we exclude those v_i nodes for which $Neighbors\ (v_i) \subseteq \cup_{j<i} Neighbors\ (v_j)$. The pseudocode is presented next:

```
Alg. 3 (G, HDS) is
  Input:  G = (N, E) - a social network;
          d - the d-hop parameter;
  Output: HDS - a d-hop dominating set for G;
  Create Gd = (N, Ed) - the d-closure of G.
  HDS = ∅;
  Order vertices in N = {v 1, v 2, … v|N|)}
    such that degree(v1) ≥ degree(v2) ≥ … ≥ degree(v|N|).
  i = 1;
  while (|∪j≤i Neighbors (vj)| < |N|) do
  if (not (Neighbors (vi) ⊆ ∪j<i Neighbors (vj)))
      HDS = HDS ∪ { vi};
    i++;
end Alg. 3.
```

III. Performance Evaluation of the Algorithms

A. Real and Synthetic Datasets

For our experiments, we used as initial networks both real and synthetic network datasets using a similar approach as in [6].

We selected seven real networks from the SNAP datasets website [11]. A short description of these seven datasets is shown in Table 1.

We generated our synthetic networks using two distribution models, namely, *Erdos-Renyi random graph model* [4] and the *configuration model* [5, 15], which generates a scale free network [7]. These types of networks are referred as ***ERNetworks*** and ***ConfNetworks*** thought this section.

TABLE I
Real Networks' Characteristics

Name	$\|\mathcal{N}\|$	$\|\mathcal{E}\|$	Description
ego-Facebook	4,039	88,234	Social circles from Facebook
email-Enron	36,692	183,831	Email communication network from Enron
ca-AstroPh	18,772	198,110	Collaboration network of Arxiv Astro Physics
ca-CondMat	23,133	93,497	Collaboration network of Arxiv Condensed Matter
ca-GrQc	5,242	14,496	Collaboration network of Arxiv General Relativity
ca-HepPh	12,008	118,521	Collaboration network of Arxiv High Energy Physics
ca-HepTh	9,877	25,998	Collaboration network of Arxiv High Energy Physics Theory

For *ERNetworks* generation we chose the following parameters. Firstly, we fixed the size of the network, $|\mathcal{N}|$, to 5,000, and we generated corresponding networks for 10 different average degree values (AVG = 2, 4, 6, ..., 20). Secondly, we selected a fixed AVG value (AVG = 10) and we varied the size of the network ($|\mathcal{N}|$ = 1K, 2K, …, 10K). For each combination of $|\mathcal{N}|$ and AVG values, we generated 5 distinct networks with those parameters.

For *ConfNetworks* generation we chose the following parameters. Firstly, we fixed the size of the network, $|\mathcal{N}|$, to 5,000, and we generated corresponding networks for 10 different power exponent values (γ = 1.7, 1.8, …, 2.6). Secondly, we selected a fixed γ value (γ = 2) and we varied the size of the network ($|\mathcal{N}|$ = 1K, 2K, …, 10K). For each combination of $|\mathcal{N}|$ and γ values, we generated 5 distinct networks.

Using the above approach we generated a total of 190 synthetic networks (95-*ERNetworks* and 95-*ConfNetworks*).

B. Experimental Results

We implemented the three algorithms described in Section II as well as the synthetic graph generators using the SNAP framework [12]. For all the experiments we used an Intel® Xeon® E5430@2.66 GHz dual CPU machine with 4 GB memory running on 32-bit Windows Server 2007 operating system. For the synthetic networks, the results shown in this section are the average results of the five synthetic datasets generated with the same parameters.

Fig. 2 – 4 present some of the results computed for real network datasets. From Fig 2 we notice that in all presented cases ***Alg. 1*** outperforms the other two algorithms in terms of 2-hop dominating size. ***Alg. 2*** and ***3*** have similar results taking turns as the second best algorithm. In terms of running time, ***Alg. 3*** is the fastest algorithm, ***Alg. 1*** is usually the second fastest, and ***Alg. 3*** is with one exception (for *ca-CondMath*) the slowest algorithm. In Fig. 3, the d-hop dominating set sizes are presented for selected networks when d parameter varies. In all instances ***Alg. 1*** determines the smallest d-hop dominating set. In general ***Alg. 3*** is the second algorithm using this comparison criterion. We do notice that the difference in dominating set sizes is greater for d = 1, and decreases for d = 2 and 3. In terms of the running time (Fig. 4), ***Alg. 3*** is the clear winner. With few exceptions, ***Alg. 1*** is also faster than ***Alg. 2***.

Fig. 5 and 6 show some of the results for *ERNetworks*. In Fig 5 the size of the initial network is kept constant (5,000), while in Fig. 6 the average degree is fixed at 10. The results for the d-hop dominating sets sizes are very similar between the two types of *ERNetworks*. In all cases the smallest dominating set size is determined by ***Alg. 1***, followed by ***Alg. 2***, and ***Alg. 3***. In terms of running time, ***Alg. 3*** is the fastest algorithm. However, ***Alg. 1*** outperforms ***Alg. 2*** for network size constant, while ***Alg. 2*** performs better when the average degree increases. Based on the fact that when the degree increases the distance between running time of ***Alg. 1*** and ***Alg. 2*** increases we conclude that average degree is the best indicator of which algorithm between ***Alg. 1*** and ***Alg. 2*** is the fastest, and ***Alg. 2*** is faster for smaller average degrees, while ***Alg. 1*** is faster for larger average degrees. Since all algorithms are applied to the d-closure network, note that the average degree of d-closure is the determining factor. Of course this value is influenced by the initial average degree.

Fig. 7 and 8 show some of the results for *ConfNetworks*. In Fig 7 the size of the initial network is kept constant (5,000), while in Fig. 6 the power exponent is fixed at 2.0. The results for the d-hop dominating sets sizes are very similar between the two types of *ERNetworks*. In terms of size, ***Alg. 1*** is followed closely by ***Alg. 3***, while ***Alg. 2*** is the clear looser. In terms of running time, ***Alg 3*** outperforms ***Alg. 2*** and ***Alg. 1***. Between the trailing algorithms, ***Alg. 1*** is faster.

Overall, ***Alg. 1*** is the best algorithm in terms of minimizing the d-hop dominating set size. ***Alg. 3*** is usually the second best algorithm in this category. The order in reversed for running time, ***Alg. 3*** having the edge over ***Alg. 1***. With several exceptions pointed out earlier, ***Alg. 2*** performs worse that ***Alg. 1*** and ***Alg. 3*** in terms of both dominating set size and running time. In conclusion, both ***Alg. 1*** and ***Alg. 3*** are good candidates for determining fast an acceptable approximation of the minimum d-hop dominating

set. Depending of which criterion is more important (size of the dominating set or running time) one of these algorithms can be selected as the best choice.

IV. Conclusions

In this paper we presented three algorithms that determine an approximation of the minimum *d*-hop dominating set for a social network. We performed extensive experiments using these three algorithms on both synthetic and real networks. We concluded that the best choice between the three algorithms is dependent of the network characteristics and which criterion is more important (size of the dominating set or running time) for the person that wants to determine *d*-hop dominating sets approximations. Regardless of these factors, the choice is between ***Alg. 1*** and ***3***, and in most cases, ***Alg. 1*** should be used when the size of the *d*-hop dominating sets is the major factor, while ***Alg. 3*** should be used when the running time is the most important factor.

As part of our future work, we intend to investigate how the proposed new algorithms can be used for more specific *d*-hop dominating set problems, such as connected *d*-hop dominating set, *d*-hop *k*-dominating set, and connected *d*-hop *k*-dominating set problems [21].

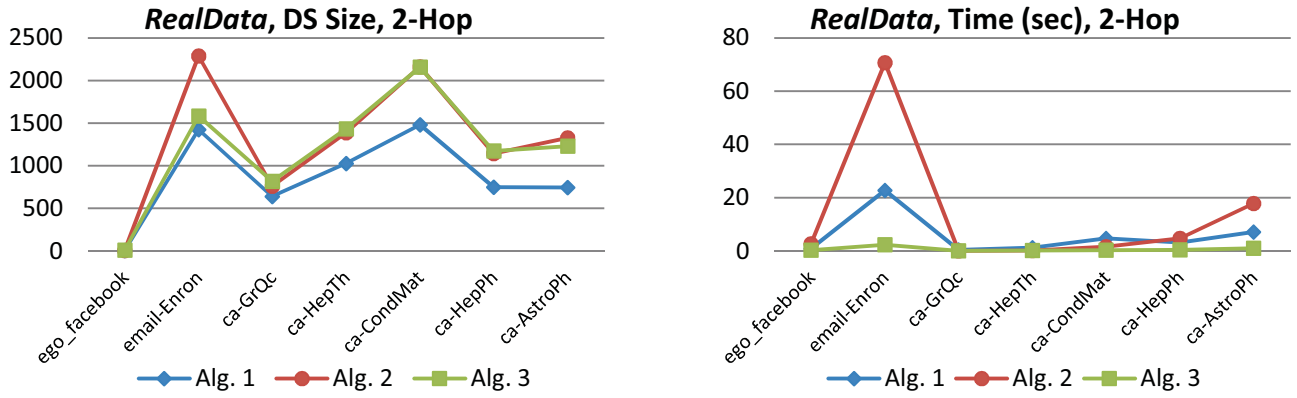


Fig. 2. The 2-hop dominating set results for real data networks. On the left the sizes of the determined 2-hop dominating sets are reported for all three algorithms. On the right, the running time of these algorithms is shown.

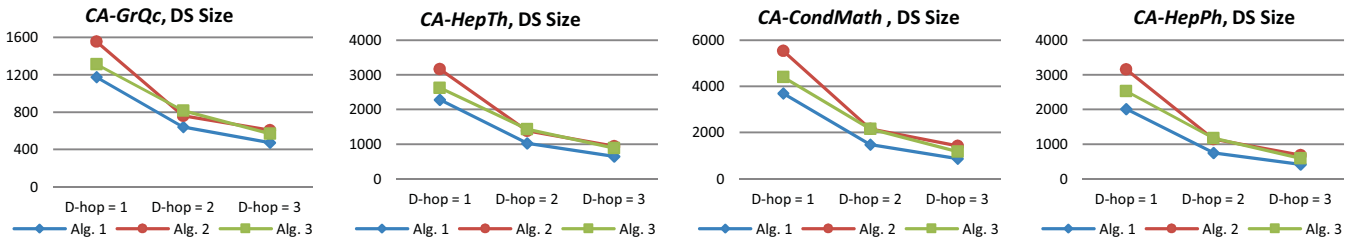


Fig. 3. The sizes of the *d*-hop dominating sets are shown for *CA-GrQc*, *CA-HepTh*, *CA-CondMath*, and *CA-HepPh* networks.

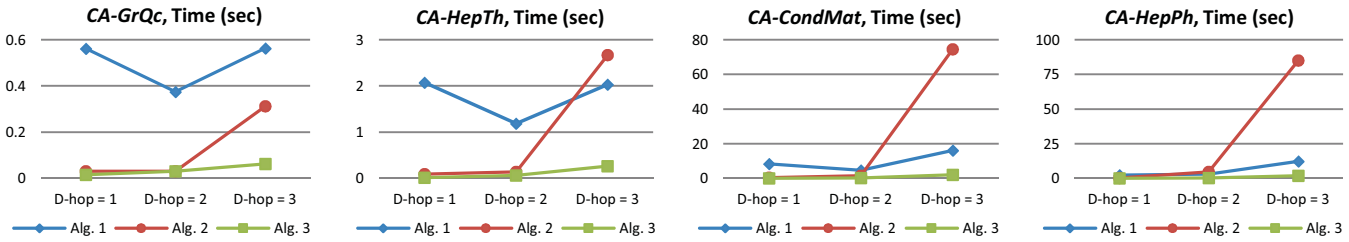


Fig. 4. The running time of running the three algorithms to determine the size of the *d*-hop dominating set for *CA-GrQc*, *CA-HepTh*, *CA-CondMath*, and *CA-HepPh* networks is shown.

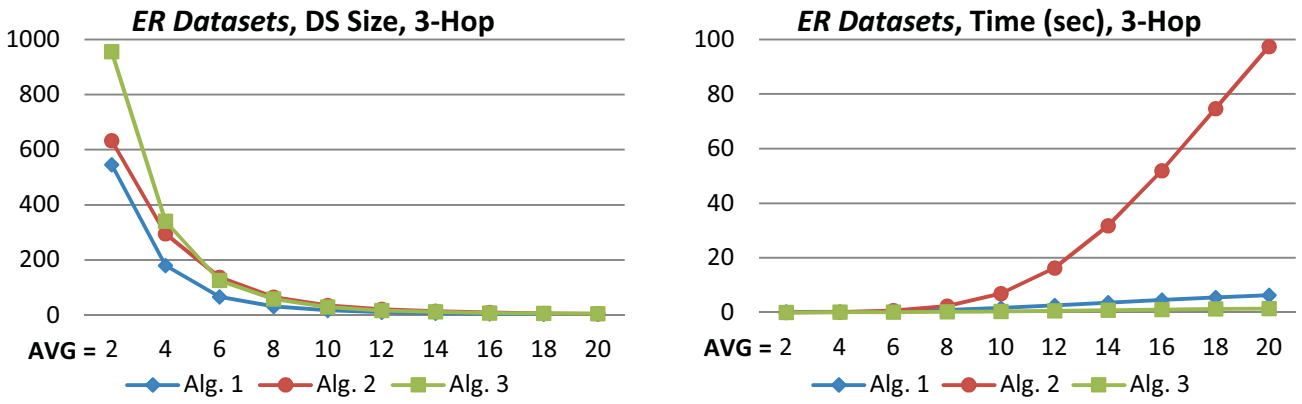


Fig. 5. The 3-hop dominating set results for *ERNetworks* when the number of nodes is constant ($\mathcal{N}$=5,000). On the left the sizes of the determined 3-hop dominating sets are reported for all three algorithms. On the right, the running time of these algorithms is shown.

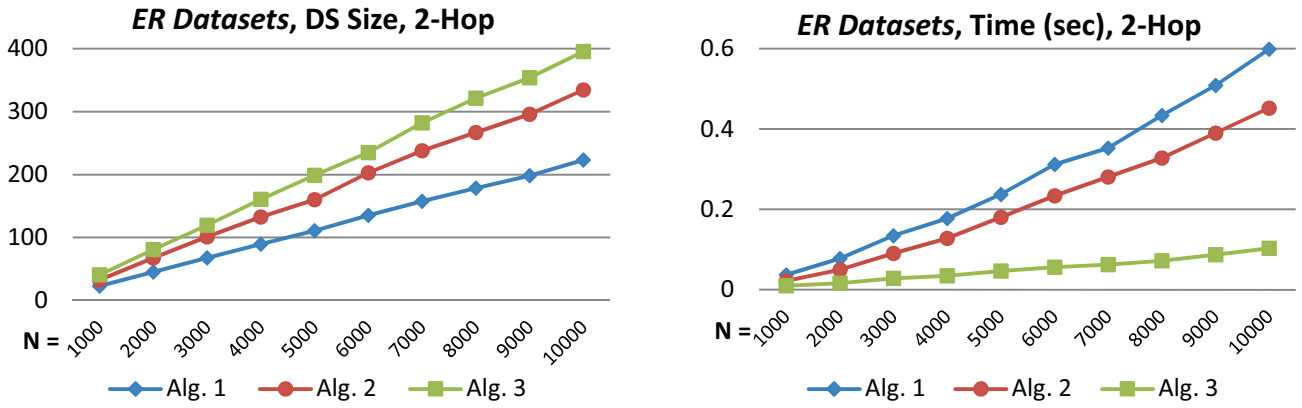


Fig. 6. The 2-hop dominating set results for *ERNetworks* when the average degree is constant (AVG = 10). On the left the sizes of the determined 2-hop dominating sets are reported for all three algorithms. On the right, the running time of these algorithms is shown.

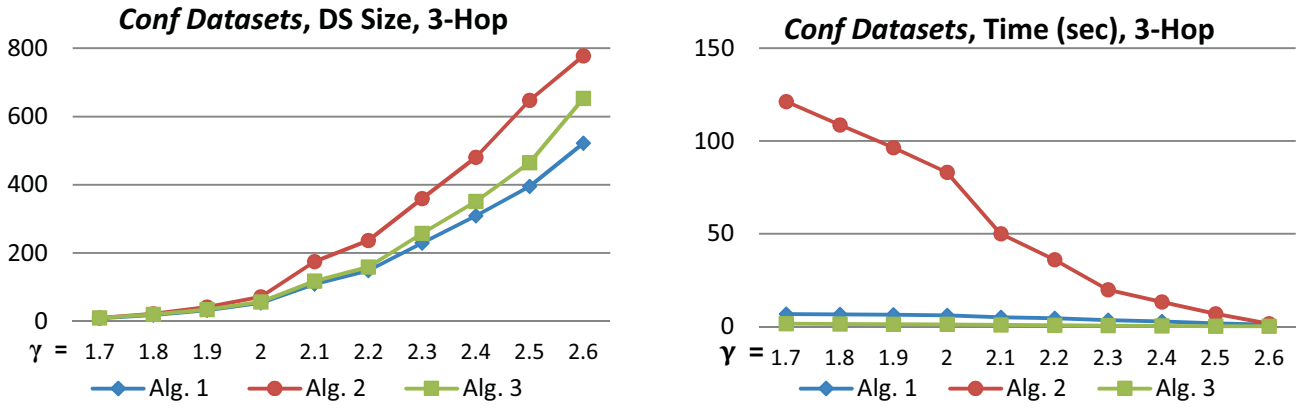


Fig. 7. The 3-hop dominating set results for *ConfNetworks* when the number of nodes is constant ($\mathcal{N}$=5,000). On the left the size of the determined 3-hop dominating set is reported for all three algorithms. On the right, the running time of these algorithms is shown.

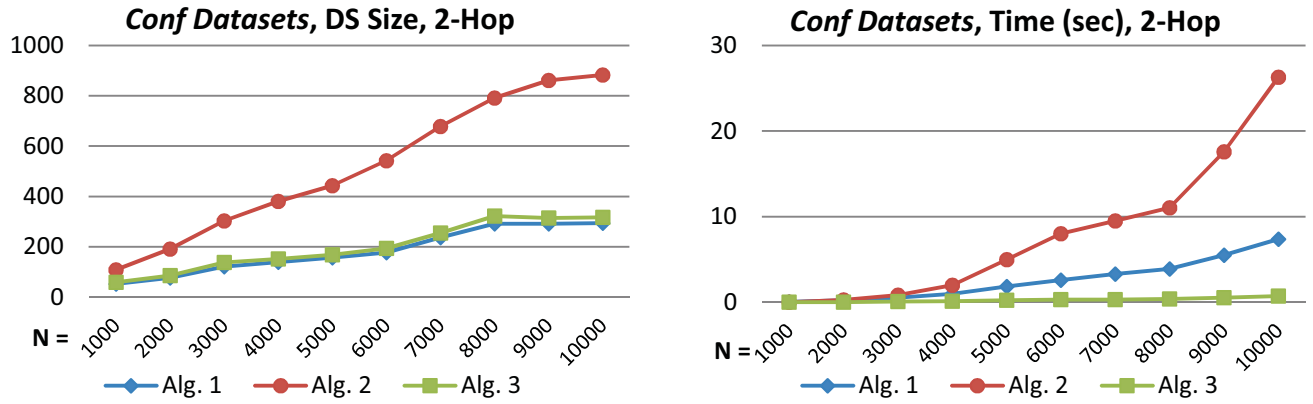


Fig. 8. The 2-hop dominating set results for *ConfNetworks* when the power exponent is constant ($\gamma = 2.0$). On the left the size of the determined 2-hop dominating set is reported for all three algorithms. On the right, the running time of these algorithms is shown.

REFERENCES

[1] A. D. Amis, R. Prakash, T. H. P. Vuong, and D. T. Huynh, "Max-min d-cluster formation in wireless ad hoc networks," In *Proc. of the Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, Vol. 1, pp. 32-41, 2000.

[2] J. Banerjee, A. Das, and A. Sen, "Analysis of d-hop dominating set problem for directed graph with indegree bounded by one," arXiv preprint arXiv:1404.6890, 2014.

[3] C. Berge, "Theory of graphs and its applications" Methuen, London, 1962.

[4] B. Bollobás, "Random graphs (2nd ed.)," Cambridge University Press. ISBN 0-521-79722-5, 2001.

[5] T. Britton, M. Deijfen, and A. Martin-Lof, "Generating simple random graphs with prescribed degree distribution," Journal of Statistical Physics, Vol. 124, Issue 6, 1377-1397, 2006.

[6] A. Campan, T. M. Truta, and M. Beckerich, "Fast dominating set algorithms for social networks," In *Proc. of the Modern Artificial Intelligence and Cognitive Science Conference (MAICS)*, pp. 55-62, 2015.

[7] M. Catanzaro, M. Boguna, and R. Pastor-Satorras, "Generation of uncorrelated random scale-free networks," Physical Review, E 71, 027103, 2005.

[8] T. N. Dinh, D. T. Nguyen, and M. T. Thai, "Cheap, easy, and massively effective viral marketing in social networks: truth or fiction?," In *Proc. of the 23rd ACM conference on Hypertext and social media*, 2012.

[9] S. Eubank, V. S. Anil Kumar, M.Marathe, A. Srinivasan, and N. Wang, "Structural and algorithmic aspects of massive social networks," In *Proc. of the ACM-SIAM Symposium on Discrete Algorithms*, pp. 718–727, 2004.

[10] M. R Garey and D. S. Johnson, "Computers and intractability: A guide to the theory of NP-completeness," W.H. Freeman, ISBN 0-7167-1045-5, page 190, 1979.

[11] J. Leskovec and A. Krevl, "SNAP datasets: Stanford large network dataset collection," 2014, Available at: http://snap.stanford.edu/data.

[12] J. Leskovec and R. Sosic, "SNAP: A general purpose network analysis and graph mining library in C++," 2014, Available at: http://snap.stanford.edu/snap.

[13] X. Li, X. Gao, and C. Zhao, "A new greedy algorithm for D-hop connected dominating set," In *Proc. of the 10th International Conference on M obile Ad-hoc and Sensor Networks (MSN)*, 2014.

[14] L. Lovasz, "On the ratio of optimal integral and fractional covers," Discrete Mathematics, Vol. 13, 383–390, 1975.

[15] M. Molloy and B. Reed, "A critical point for random graphs with a given degree sequence," Random Structures and Algorithms, Vol. 6, 161-180, 1995.

[16] F. Molnar, S. Sreenivasan K. Szymanski, and G. Korniss, "Minimum dominating sets in scale-free network ensembles," Scientific Reports, No. 3, Article number 1736, Nature Publishing Group, 2013.

[17] J. C. Nacher and T. Akutsu, "Dominating scale-free networks with variable scaling exponent: Heterogeneous networks are not difficult to control," New Journal of Physics, Vol. 14, 2012.

[18] O. J. Reuter and D. Szakonyi, "Online social media and political awareness in authoritarian regimes," Working Paper, Series: Political Science, WP BRP 10/PS/2012, National Research University, 2012, Available online at: https://iims.hse.ru/data/2013/01/18/1305921529/10PS2012.pdf .

[19] M. Q. Rieck, S. Pai, and S.Dhar, "Distributed routing algorithms for multi-hop ad hoc networks using d-hop connected d-dominating sets," Computer Networks, Vol. 47, No. 6, pp. 785-799, 2005.

[20] S. Whitten, "Can 140 characters affect the 2016 presidential election?," in CNBC, 2015, Available online at: http://www.cnbc.com/2015/11/05/can-140-characters-affect-the-2016-presidential-election.html.

[21] J. Yu, N. Wang, G. Wang, and D. Yu, "Connected dominating sets in wireless ad hoc and sensor networks – A comprehensive survey," Computer Communications, Vol. 36, No. 2, pp. 121-134, 2013.

Self-Organizing Map Convergence

Robert Tatoian **Lutz Hamel**
Department of Computer Science and Statistics
University of Rhode Island
Kingston, Rhode Island, USA
hamel@cs.uri.edu

Abstract

Self-organizing maps are artificial neural networks designed for unsupervised machine learning. They represent powerful data analysis tools applied in many different areas including areas such as biomedicine, bioinformatics, proteomics, and astrophysics. We maintain a data analysis package in R based on self-organizing maps. The package supports efficient, statistical measures that enable the user to gauge the quality of a generated map. Here we introduce a new quality measure called the convergence index. The convergence index is a linear combination of map embedding accuracy and estimated topographic accuracy and since it reports a single statistically meaningful number it is perhaps more intuitive to use than other quality measures. Here we study the convergence index in the context of clustering problems proposed by Ultsch as part of his fundamental clustering problem suite. We demonstrate that the convergence index captures the notion that a SOM has learned the multivariate distribution of a training data set.

1 Introduction

Self-organizing maps are artificial neural networks designed for unsupervised machine learning. They represent powerful data analysis tools applied in many different areas including areas such as biomedicine, bioinformatics, proteomics, and astrophysics [1]. We maintain a data analysis package in R called `popsom` [2] based on self-organizing maps. The package supports efficient, statistical measures that enable the user to gauge the quality of a generated map [3]. Here we introduce a new quality measure called the *convergence index*. The convergence index is a linear combination of map embedding accuracy and estimated topographic accuracy. It reports a single, statistically meaningful number between 0 and 1 – 0 representing a least fitted model and 1 representing a completely fitted model – and is therefore perhaps more intuitive to use than other quality measures. Here we study the convergence index in the context of clustering problems proposed by Ultsch as part of his fundamental clustering problem suite [4]. In particular, we are interested in how well the convergence index captures the notion that a SOM has learned the multivariate distribution of a training data set.

Over the years a number of different quality measures for self-organizing maps have been proposed. Nice overviews of common SOM quality measures appear in [5] and [6]. Our convergence index distinguishes itself from many of the other measures in that it is statistical in nature. This is particularly true for the part of the convergence index based on embedding (or coverage) which is essentially a two-sample test between the training data and the set of self-organizing map neurons viewed as populations. The two sample test measures how similar these two populations are. For a fully embedded map the population of neurons should be indistinguishable from the training data. This statistical view of embedding is interesting because it makes the standard visualization of SOMs using a U-matrix [7] statistically meaningful. That is, the cluster and the distance interpretations of the U-matrix now have a statistical foundation based on the fact that the distribution of the map neurons is indistinguishable from the distribution of the training data.

The other part of our convergence index, the estimated topographic accuracy, is an efficient statistical approach to the usual topographic error quality measure [5] which can be computationally expensive. In our approach we use a sample of the training data to estimate the topographic accuracy. Experiments have shown that we need only a fairly small sample of the training data to get very accurate estimates.

The remainder of this paper is structured as follows. In Section 2 we briefly review self-organizing maps as implemented by our package. We take a look at the quality measures implemented in `popsom` in Section 3. In Section 4 we define our convergence index. We discuss our case studies in Section 5. Conclusions and further work are discussed in Section 6

2 Self-Organizing Maps

Briefly, a self-organizing map [1] is a kind of artificial neural network that implements competitive learning, which can

ISBN: 1-60132-431-6, CSREA Press ©

be considered a form of unsupervised learning. On the map itself, neurons are arranged along a rectangular grid with dimensions x_{dim} and y_{dim}. Learning proceeds in two steps for each training instance $\vec{x}_k$, $k = 1, 2, 3, \ldots, M$, with M the number of training instances:

1. The **competitive step** where the best matching neuron for a particular training instance is found on the rectangular grid,

$$c = \underset{i}{\operatorname{argmin}}(||\vec{m}_i - \vec{x}_k||)$$

where $i = 1, 2, \ldots, N$ is an index over the neurons of the map with $N = x_{dim} \times y_{dim}$ is the number of neurons on the grid, and $\vec{m}_i$ is a neuron indexed by i. Finally, c is the index of the best matching neuron $\vec{m}_c$ on the map.

2. The **update step** where the training instance $\vec{x}_k$ influences the best matching neuron $\vec{m}_c$ and its neighborhood. The update step can be represented by the following update rule for the neurons on the map,

$$\vec{m}_i \leftarrow \vec{m}_i - \eta\vec{\delta}_i h(c, i)$$

for $i = 1, 2, \ldots, N$. Here $\vec{\delta}_i = \vec{m}_i - \vec{x}_k$, η is the learning rate, and $h(c, i)$ is a loss function with,

$$h(c, i) = \begin{cases} 1 & \text{if } i \in \Gamma(c), \\ 0 & \text{otherwise,} \end{cases}$$

where $\Gamma(c)$ is the neighborhood of the best matching neuron $\vec{m}_c$ with $c \in \Gamma(c)$. Typically the neighborhood is a function of time and its size decays during training. Initially the neighborhood for neuron $\vec{m}_c$ includes all other neurons on the map,

$$\Gamma(c)|_{t=0} = \{1, 2, \ldots, N\}.$$

As training proceeds the neighborhood for $\vec{m}_c$ shrinks down to just the neuron itself,

$$\Gamma(c)|_{t\gg 0} = \{c\}.$$

Here, as before, $N = x_{dim} \times y_{dim}$ is the number of neurons on the map. This means that initially the update rule for each best matching neuron has a very large field of influence which gradually shrinks to the point that the field of influence just includes the best matching neuron itself.

The two training steps above are repeated for each training instance until the given map converges.

Figure 1 shows a scatter plot of the Tetra problem in Ultsch's fundamental clustering problem suite (FCPS) [4]. The data set consists of four almost touching clusters embedded in three dimensional space. Figure 2 shows a SOM starburst plot of this data set generated with our `popsom` package [2] which supports statistical convergence criteria [3] and

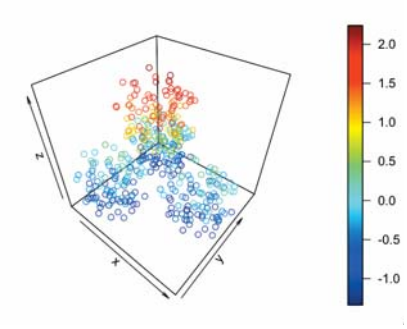


Figure 1: The Tetra data set.

detailed cluster visualizations in terms of our starburst plots [8].

The four clusters can easily be identified on the map by their starbursts. Also easily visible is the fact that clusters themselves are identified by their light color and cluster boundaries are identified by darker colors. The easily identified borders mean that the clusters are indeed distinct clusters. Their relative position is also meaningful to a point, given that this is a 2D rendering of a higher dimensional space. All these observations are justified due to the fact that the map has converged and therefore positioning and distance amongst clusters is statistically meaningful.

3 Quality Measures

3.1 Map Embedding Accuracy

Yin and Allinson have shown that under some mild assumptions the neurons of a large enough self-organizing map will converge on the probability distribution of the training data given infinite time [9]. This is the motivation for our map embedding accuracy:

> *A SOM is completely embedded if its neurons appear to be drawn from the same distribution as the training instances.*

This view of embedding naturally leads to a two-sample test [10]. Here we view the training data as one sample from some probability space $\mathbf{X}$ having the probability density function $p(x)$ and we treat the neurons of the SOM as another sample. We then test to see whether or not the two samples appear to be drawn from the same probability space. If we operate under the simplifying assumption that each of the d features of the input space $\mathbf{X} \subset \mathbb{R}^d$ are normally distributed and independent of each other, we can test each of the features separately. This assumption leads to a fast algorithm for identifying SOM embedding: We define a feature as embedded if the variance and the mean of that feature appear to be drawn from the same distribution for both the training data and the

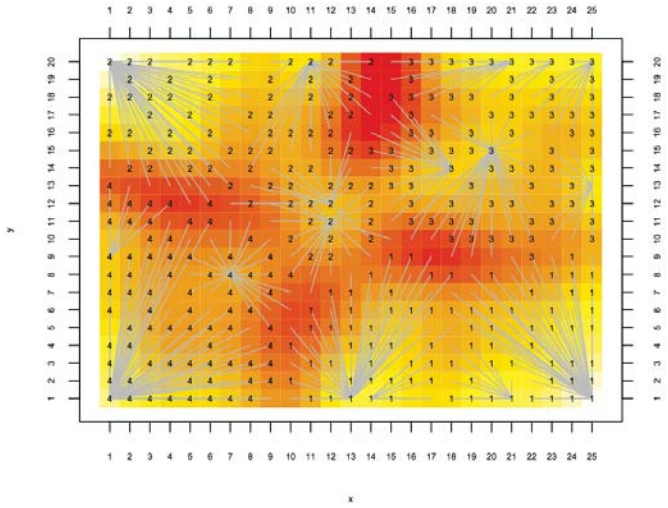


Figure 2: A SOM starburst plot of the Tetra data set.

neurons. If all the features are embedded then we say that the map is *fully embedded*.

The following is the formula for the $(1-\alpha)*100\%$ confidence interval for the ratio of the variances from two random samples [10],

$$\frac{s_1^2}{s_2^2}\cdot\frac{1}{f_{\frac{\alpha}{2},n_1-1,n_2-1}} < \frac{\sigma_1^2}{\sigma_2^2} < \frac{s_1^2}{s_2^2}\cdot f_{\frac{\alpha}{2},n_1-1,n_2-1}, \qquad (1)$$

where s_1^2 and s_2^2 are the values of the variance from two random samples of sizes n_1 and n_2 respectively, and where $f_{\frac{\alpha}{2},n_1-1,n_2-1}$ is an F distribution with n_1-1 and n_2-1 degrees of freedom. To test for SOM embedding, we let s_1^2 be the variance of a feature in the training data and we let s_2^2 be the variance of that feature in the neurons of the map. Furthermore, n_1 is the number of training samples and n_2 is the number of neurons in the SOM. The variance of a particular feature of both training data and neurons appears to be drawn from the same probability space if 1 lies in the confidence interval denoted by equation (1): the ratio of the underlying variance as modeled by input space and the neuron space, respectively, is approximately equal to one, $\sigma_1^2/\sigma_2^2 \approx 1$, up to the confidence interval.

In the case where $\bar{x}_1$ and $\bar{x}_2$ are the values of the means from two random samples of size n_1 and n_2, and the variances of these samples are σ_1^2 and σ_2^2 respectively, the following formula provides $(1-\alpha)*100\%$ confidence interval for the difference between the means [10],

$$\mu_1-\mu_2 \;>\; (\bar{x}_1-\bar{x}_2) - z_{\frac{\alpha}{2}}\cdot\sqrt{\frac{\sigma_1^2}{n_1}+\frac{\sigma_2^2}{n_2}}, \qquad (2)$$

$$\mu_1-\mu_2 \;<\; (\bar{x}_1-\bar{x}_2) + z_{\frac{\alpha}{2}}\cdot\sqrt{\frac{\sigma_1^2}{n_1}+\frac{\sigma_2^2}{n_2}}. \qquad (3)$$

The mean of a particular feature for both training data and neurons appears to be drawn from the same probability space if 0 lies in the confidence interval denoted by equations (2) and (3). Here $z_{\frac{\alpha}{2}}$ is the appropriate z score for the chosen confidence interval.

We say that a feature is embedded if the above criteria for both the mean and variance of that feature are fulfilled. We can now define the *map embedding accuracy* for d features,

$$ea = \frac{1}{d}\sum_{i=1}^{d}\rho_i, \qquad (4)$$

where

$$\rho_i = \begin{cases} 1 & \text{if feature } i \text{ is embedded,} \\ 0 & \text{otherwise.} \end{cases}$$

The map embedding accuracy is the fraction of the number of features which are actually embedded (i.e. those features whose mean and variance were adequately modeled by the neurons in the SOM). With a map embedding accuracy of 1 a map is fully embedded. In order to enhance the map embedding accuracy in our implementation [2], we multiply each embedding term ρ_i by the significance of the corresponding feature i which is a Bayesian estimate of that feature's relative importance [11]. A more in-depth statistical analysis of our map embedding accuracy can be found in [12].

3.2 Estimated Topographic Accuracy

Many different approaches to measuring the topological quality of a map exist, *e.g.* [13, 14]. But perhaps the simplest measure of the topological quality of a map is the *topographic error* [15] defined as:

$$te = \frac{1}{n}\sum_{i=1}^{n} err(\vec{x}_i) \tag{5}$$

with

$$err(\vec{x}_i) = \begin{cases} 1 & \text{if } bmu(\vec{x}_i) \text{ and } 2bmu(\vec{x}_i) \text{ are not neighbors,} \\ 0 & \text{otherwise.} \end{cases}$$

for training data $\{\vec{x}_1, \ldots, \vec{x}_n\}$ where $bmu(\vec{x}_i)$ and $2bmu(\vec{x}_i)$ are the best matching unit and the second-best matching unit for training vector $\vec{x}_i$ on the map, respectively. We define the *topographic accuracy* of a map as,

$$ta = 1 - te. \tag{6}$$

Computing the topographic accuracy can be very expensive, especially for large training data sets and/or maps. One way to ameliorate the situation is to sample the training data and use this sample S to estimate the topographic accuracy. If we let s be the size of the sample then the *estimated topographic accuracy* is,

$$ta' = 1 - \frac{1}{s}\sum_{\vec{x}\in S} err(\vec{x}). \tag{7}$$

We have shown in [3] that we can get accurate values for ta' with very small samples.

In addition to computing the value for the estimated topographic accuracy we use the bootstrap [16] to compute values for an appropriate confidence interval in order to give us further insight into the estimated topographic accuracy in relation to the actual value for the topographic accuracy whose value should fall within the bootstrapped confidence interval.

It is easy to see from (7) that for topologically faithful maps the estimated topographic accuracy should be close to 1. We then say that the map is *fully organized*.

4 The Convergence Index

Recently it has been argued that any SOM quality measure needs to report on both the embedding of a map in the input data space as well as the topological quality of a map [17]. In order to have a simple interpretation of the embedding and the topographic accuracy we introduce the convergence index, cix, which is a linear combination of the two quality measures introduced in the previous section,

$$cix = \frac{1}{2}ea + \frac{1}{2}ta'. \tag{8}$$

Table 1: Training results for the Tetra data set.

$iter$	cix	ea	ta'	$(lo\text{-}hi)$
1	0.01	0.00	0.02	(0.00 - 0.06)
10	0.02	0.00	0.04	(0.00 - 0.10)
100	0.35	0.00	0.70	(0.58 - 0.82)
1000	0.61	0.33	0.88	(0.80 - 0.96)
10000	0.99	1.00	0.98	(0.94 - 1.00)
100000	1.00	1.00	1.00	(1.00 - 1.00)

The convergence index is equal to 1 for a fully embedded and fully organized map. In our previous studies [3, 12] we have shown that the embedding accuracy and the estimated topographic accuracy are conservative measures and therefore subsume many of the other SOM quality measures.

5 Case Studies

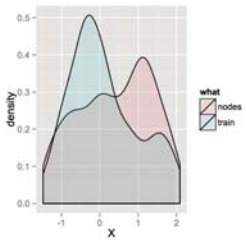
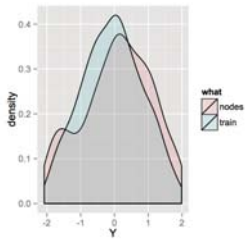
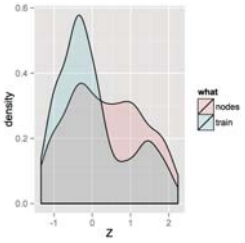
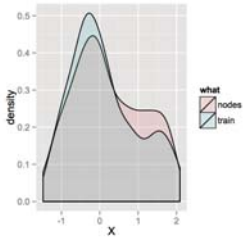
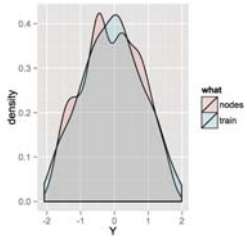
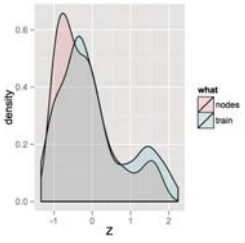
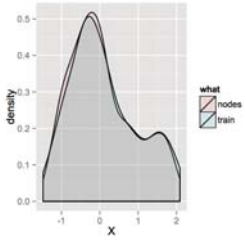
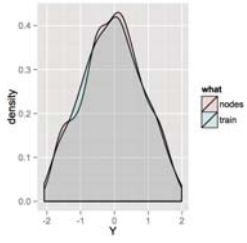
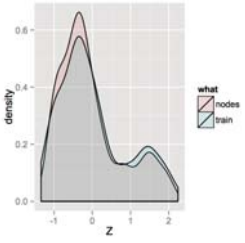
We apply our SOM algorithm to two data sets from Ultsch's fundamental clustering problem suite (FCPS) [4]. We are particularly interested in how the distribution of the neurons converges on the distribution of the training data set as training of the map progresses and how this relates to our convergence index. Our experiments seem to confirm that when $cix \approx 1$ then the distribution of the neurons matches the distribution of the training data almost exactly as measured on the marginals.

5.1 The Tetra Data Set

Our first data set is the Tetra data set. As with all the data sets in FCPS this data set is a synthetic data set with four almost touching clusters in three dimensions. Figure 1 shows a scatter plot of this data set. The four clusters are clearly identifiable. We trained a 25×20 SOM using this data set stepping the training iterations in powers of 10 from 1 to 100000. Table 1 shows the results. We can observe that the convergence index (cix) starts from just about 0 and grows to 1. Also included in the table are the embedding accuracy (ea) and the estimated topographic accuracy (ta') together with its 95% confidence interval ($lo\text{-}hi$).

In Table 2 we show the distribution of the neurons (pink) compared to the distribution of the training data (green) in relation to the cix and the number of training iterations $iter$. We show the distributions of the three marginals X, Y, and Z at training iterations 10, 1000, and 100000, respectively. It is striking to see how precisely the neurons model the training data distribution when $cix = 1$. At least in the case of the X and Y marginals. In the Z marginal we can see some discrepancy and we are wondering if that is due to the fact that we are making the simplifying assumption of a normal distribution when testing for embedding accuracy. One of our research

Table 2: Distribution of the neurons and training data (Tetra data set).

iter	cix	X	Y	Z
10	0.01			
1000	0.61			
100000	1.00			

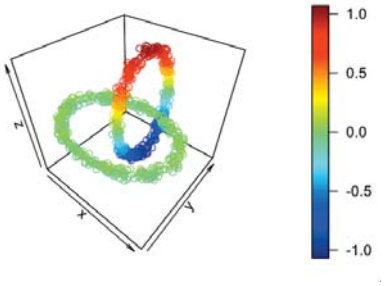


Figure 3: The Chainlink data set.

Table 3: Training results for the Chainlink data set.

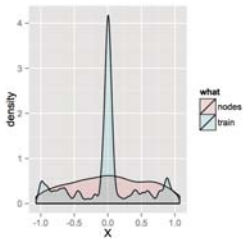
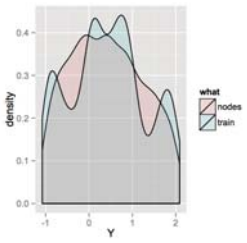
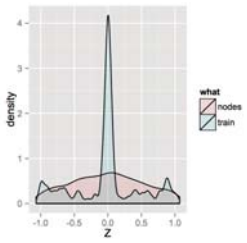
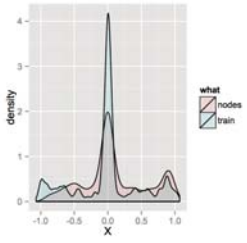
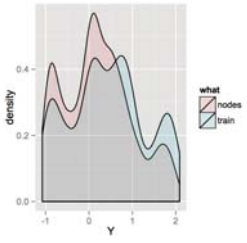
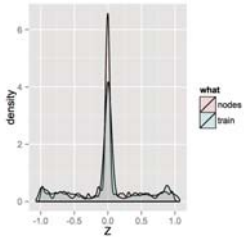
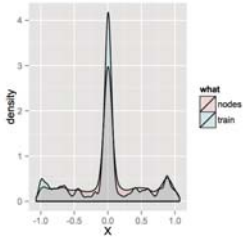
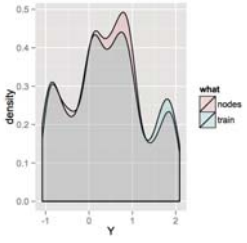
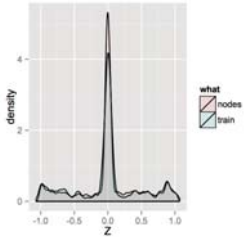
$iter$	cix	ea	ta'	$(lo\text{-}hi)$
1	0.31	0.61	0.00	(0.00 - 0.00)
10	0.33	0.61	0.04	(0.00 - 0.10)
100	0.12	0.00	0.24	(0.14 - 0.38)
1000	0.39	0.00	0.78	(0.66 - 0.90)
10000	0.57	0.19	0.94	(0.84 - 1.00)
100000	0.89	0.81	0.96	(0.90 - 1.00)

goals is to replace the parametric embedding test with a distribution free test. Figure 2 shows the resulting SOM starburst of the Tetra set after 100000 training iterations. As expected, the clusters are well defined and separated given $cix = 1$.

5.2 The Chainlink Data Set

The second data set we look at is the Chainlink data set. The interesting aspect of this data set is that the two classes defined by this data set are not linearly separable. Figure 3 shows a scatter plot of this data set. We proceeded in the same way as we did for the first data set in that we trained a 45×40 SOM in powers of 10 from 1 to 100000. Table 3 shows the results of this training. What is perhaps striking is that the embedding accuracy ea is fairly large initially and then drops off to 0 only to then increase back to something close to 1. A look at Table 4 perhaps explains this in that the randomly generated initial population of neurons happens to model the distribution of the Y marginal quite well even without a lot of training. Notice however that the estimated topographic accuracy for these initial maps is equal to zero. That means this distribution is generated by neurons that are not properly organized on the map. This explains why once

Table 4: Distribution of the neurons and training data (Chainlink data set).

iter	cix	X	Y	Z
10	0.33			
1000	0.39			
100000	0.89			

a larger number of training iterations is applied to the map the embedding accuracy drops down to 0 before properly organized neurons model the distribution. At 100000 iterations we have $cix = 0.89$ and we can see that the distributions on the marginals are modeled quite well. However, for this non-separable data set we do not expect to reach a convergence index of 1 since from a topological point of view it is very to difficult to have a sheet model two interlocking rings.

Figure 4 shows the SOM starburst plot of the Chainlink data set after 100000 iterations. The clusters are well defined and separated with one exception. As suspected, given a convergence index of less than 1, we have a strange fold in the center of the map where the SOM tried to accommodate the interlocking rings. In our experience, even when training is restarted with a different set of initial conditions some sort of anomaly will always emerge while the SOM tries to accommodate the interlocking rings. These anomalies prevent the convergence index to become 1. In some sense this is satisfying, since now we can view the convergence index as a way to also measure the difficulty of the input space.

6 Conclusions and Further Work

Self-organizing maps are powerful data analysis tools applied to many different areas such as biomedicine, genomics, and physics. We maintain a data analysis package in R based on self-organizing maps. The package supports efficient, statistical measures enabling the user to gauge the quality of a given map. Here we introduced a new quality measure called the convergence index and demonstrated that it captures the notion that a SOM has learned the multivariate distribution of a training data set. The advantages of this new quality measure is that it is intuitive and easy to interpret.

Because the SOM algorithm is a constrained learner in the sense that neurons are not able to freely move around the space spanned by the training data it is sufficient to test the marginals for convergence. Our next step is to make this argument statistically rigorous. We also would like to dispense with the normality assumptions in our tests. The one-dimensional Kolmogorov-Smirnov test [18] seems to be suitable here.

References

[1] T. Kohonen, *Self-organizing maps*. Springer Berlin, 2001.

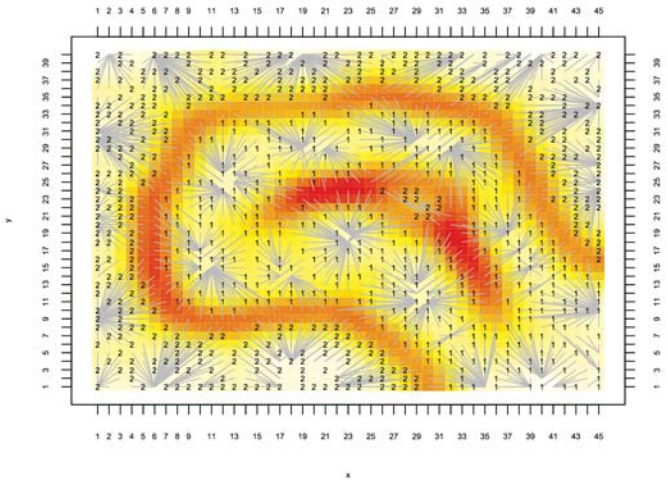


Figure 4: A SOM starburst plot of the Chainlink data set.

[2] L. Hamel, B. Ott, and G. Breard, *popsom: Self-Organizing Maps With Population Based Convergence Criterion*, 2015. R package version 3.0.

[3] L. Hamel, "Som quality measures: An efficient statistical approach," in *Advances in Self-Organizing Maps and Learning Vector Quantization*, pp. 49–59, Springer, 2016.

[4] A. Ultsch, "Clustering wih som: U* c," in *Proceedings of the 5th Workshop on Self-Organizing Maps*, vol. 2, pp. 75–82, 2005.

[5] G. Pölzlbauer, "Survey and comparison of quality measures for self-organizing maps," in *Proceedings of the Fifth Workshop on Data Analysis (WDA-04)*, pp. 67–82, Elfa Academic Press, 2004.

[6] R. Mayer, R. Neumayer, D. Baum, and A. Rauber, "Analytic comparison of self-organising maps," in *Advances in Self-Organizing Maps*, pp. 182–190, Springer, 2009.

[7] A. Ultsch, "Self organized feature maps for monitoring and knowledge aquisition of a chemical process," in *ICANN'93*, pp. 864–867, Springer, 1993.

[8] L. Hamel and C. W. Brown, "Improved interpretability of the unified distance matrix with connected components," in *7th International Conference on Data Mining (DMIN'11)*, pp. 338–343, 2011.

[9] H. Yin and N. M. Allinson, "On the distribution and convergence of feature space in self-organizing maps," *Neural computation*, vol. 7, no. 6, pp. 1178–1187, 1995.

[10] I. Miller and M. Miller, *John E. Freund's Mathematical Statistics with Applications (7th Edition)*. Prentice Hall, 7 ed., 2003.

[11] L. Hamel and C. W. Brown, "Bayesian probability approach to feature significance for infrared spectra of bacteria," *Applied Spectroscopy*, vol. 66, no. 1, pp. 48–59, 2012.

[12] L. Hamel and B. Ott, "A population based convergence criterion for self-organizing maps," in *Proceedings of the 2012 International Conference on Data Mining*, (Las Vegas, Nevada), July 2012.

[13] E. Merényi, K. Tasdemir, and L. Zhang, "Learning highly structured manifolds: harnessing the power of SOMs," in *Similarity-based clustering*, pp. 138–168, Springer, 2009.

[14] T. Villmann, R. Der, M. Herrmann, and T. M. Martinetz, "Topology preservation in self-organizing feature maps: exact definition and measurement," *Neural Networks, IEEE Transactions on*, vol. 8, no. 2, pp. 256–266, 1997.

[15] K. Kiviluoto, "Topology preservation in self-organizing maps," in *IEEE International Conference on Neural Networks*, pp. 294–299, IEEE, 1996.

[16] B. Efron and R. J. Tibshirani, *An introduction to the bootstrap*. CRC press, 1994.

[17] D. Beaton, I. Valova, and D. MacLean, "Cqoco: A measure for comparative quality of coverage and organization for self-organizing maps," *Neurocomputing*, vol. 73, no. 10, pp. 2147–2159, 2010.

[18] R. Wilcox, "Kolmogorov–smirnov test," *Encyclopedia of biostatistics*, 2005.

A probabilistic logic-based approach for subjective interestingness analysis

José Carlos Ferreira da Rocha, Alaine Margarete Guimarães and Valter Luis Estevam Jr

Abstract— **This paper describes an approach that uses probabilistic logic reasoning to compute subjective interestingness scores for classification rules. In the proposed approach domain knowledge is represented as a probabilistic logic program which encodes information incoming from experts and statistical data. Computation of interestingness scores is supported by a reasoning procedure that uses linear programming to calculate the probabilities of interest. An application example illustrates the utilization of the described approach in evaluating classification rules on UCI Wisconsin Cancer data set. As it is shown this scheme provides a mechanism to estimate probability based subjective interestingness scores.**

I. Introduction

Searching patterns in databases has been a useful strategy for acquiring the knowledge that is required to perform tasks involving decision making and problem-solving in areas such as medicine, agriculture, biology, environmental research and many others. In this context, knowledge discovery in data bases (KDD), the field of computer science, intends to develop the theoretical basis and computational frameworks to transform raw data into useful and comprehensive information. In practice, KDD process can be abstracted in three basic steps: preprocessing data to make it ready for analysis, running data mining algorithms to find out relationships among the variables of the application domain and then, evaluating how interesting are the detected patterns.

A common application in KDD is mining classification rules that aiming to finding logical implications which relate the features of an object to a label that informs its category (or class) by examining cases in a data set. The main criteria for evaluating rule mining results is the accuracy, the overall correctness of the model in predicting object class in a test data set. Basically, if the discovered rules are sufficiently accurate it is possible to use them to classify new object instances given the evidence.

However, depending on application objectives, accuracy is not the only relevant criterion in classification rule mining. In some situations, it might also be important to estimate the usefulness, consistency or novelty of the learned patterns in the light of domain knowledge. It is the aim of subjective interestingness analysis, a KDD procedure that explores domain knowledge to calculate scores that try to quantify how much a pattern meets a criterion of interest. For example, by verifying if the obtained rules are consistent with the background knowledge [1], [2]. - that is, if they are expected given the available information.

To perform this task, data analysts usually make use of a knowledge-based system which provides the functionalities required to compare the data mining results to previous expectations. Such approach demands the implementation of a knowledge base that encodes all relevant information and thus, involves carrying out a knowledge engineering process to acquire the background information and then writing the elicited sentences using an appropriated formalism. However, domain knowledge is, frequently, incomplete and imprecise and, in this case, the selected formalism must provide the mechanism to proceed reasoning under uncertainty.

Probability theory has been widely employed to represent uncertain beliefs about the statements in a knowledge base and perform inferences about expectations. However, as observed by Walley [3] [4], domain experts do not always feel comfortable to assign exact probabilities to the sentences of a knowledge base. In other terms, sometimes it can be the case that the information provided by the experts is not enough to the elicitation of point probability estimates. Since this situation can also arise during the development of a knowledge base for subjective interestingness analysis, the use formalisms which allow to handle with imprecise probability statements can be very useful in practice.

Considering it, this work presents an approach that explores the formalism of probabilistic logic to encode the prior knowledge and uses a linear programming-based procedure to compute interestingness scores for classification rules. The main objective underlying the proposal is to provide a basic framework which allows to integrate domain knowledge from different sources and also to execute the needed inferences. In particular, the proposed approach is able to deal with information from probabilistic distributions defined on the attributes which appear in a classification rule, statistics about correlation data and imprecise beliefs elicited from experts.

The paper is organized as follows. Section 2 brings a review on classification rule mining and an interestingness measure called level, proposed by Gay and M. Boullé [5]. This section also describes some concepts in probabilistic logic. Section 3 presents the proposed approach. Section 4 illustrates the utilization of the described approach with a simple application in which the level of interest of classification rules generated by the JRIP algorithm on the UCI Breast Cancer Data Set is calculated. Section 5 discusses the main issues related to the employment of the proposed approach in interestingness analysis. The last section presents the final

J.C.F. da Rocha is with the Computational Intelligence Lab, UEPG, Ponta Grossa, PR 84030-900, Brazil (email: jrocha@uepg.br).

A.M.Guimarães is with the Infoagro Lab, UEPG, Ponta Grossa, PR 84030-900, Brazil (email: alainemg@hotmail.com).

Valter L. Estevam Jr is with Department of Informatics at IFC, Videira, SC 84030-900, Brazil.

remarks of this study.

II. Background review

Let $\mathbf{D}$ be a multivariate data set with m instances, n descriptor attributes and a target attribute. Denote the descriptors by $X_1 ..., X_n$ and the target or class by C. Each attribute X_j symbolizes some feature of objects to be classified and its domain[1] is denoted as Ω_j. The target attribute is a category label.

Classification rule mining aims at discovering implication patterns that can be used to classify objects into given categories based on their features [6]. It is the task of inducing, from $\mathbf{D}$, logical expressions of the form $F_1 \wedge F_2 \cdots \wedge F_t \rightarrow C$. In this work, each antecedent F_i of a rule is assumed to be a relational expression $X_j \odot x_{j,k}$ where $x_{j,k} \in \Omega_j$, $\odot$ is an operator from the set $\{<, >, \leq, \geq, =\}$ and $t \in \mathbb{N}^+$ with $t \leq n$. The consequent can be viewed as a proposition labeling an object as a member of a particular class. There is a finite number of classes.

As it does for other data mining procedures used in KDD, it is necessary to verify whether the rule mining results meet the application requirements. In this kind of analysis, called interestingness analysis, some numerical scores, the interestingness measures, are computed for each discovered pattern. Good interestingness measures should indicate how much a pattern complies with KDD goals by identifying "valid, novel, potentially useful and ultimately understandable information in data" [7]. As there are many different interestingness scores in literature it is necessary to choose a suitable one for each application. Afterward, the selected score can be used to filter or rank the most promising results.

Since subjective interestingness analysis explores domain knowledge and user preferences to rank the discoveries, its realization requires building a knowledge base as well as providing the associated reasoning procedures. Furthermore, its implementation often faces some issues related to the development of knowledge-based systems. Two of them are how to deal with uncertain and incomplete knowledge [8], [9]. Probability theory has been a ubiquitous tool to handle these conditions and considering it, D. Gay and M. Boullé [5] proposed an interestingness score, named level, which is defined as follows:

$$level(R) = 1 - \frac{c(R)}{c(R_0)} \tag{1}$$

In this expression, $c(R) = -log(P(\mathbf{D}|R)) - log(P(R))$ is said to be the cost of the rule and $c(R_0)$ is the cost of a default rule - it is computed from class frequencies in the data set. As it can be seen, $level(R)$ is a posterior probability-based score and as such evaluates, simultaneously, whether a rule fits to data and agrees with prior knowledge. If $level(R) \leq 0$, the rule is less than or as probable as the default rule and it is said uninteresting. If $0 < level(R) < 1$, R is more probable than the default rule and it is more interesting as its level approaches 1. If $level(R) = 1$, the rule fits the observations and prior beliefs exactly.

A point to be noted here is that eliciting probabilities from experts is difficult activity [10]. So, if previous data or literature provides information that can be relevant for interestingness analysis (probability distributions, descriptive statistics, correlation), it should be integrated into the knowledge base whenever it is possible. Another point is that eliciting probabilities from experts or literature does not always manage to obtain exact probability assignments. In particular, in some domains, it can be the case that all available information is formed by imprecise probabilities [3] or qualitative beliefs [11]. From this follows that it can be useful to select formalisms which are is able to deal with numeric and interval-valued probabilities and qualitative probabilities [12] [13].

A. Propositional probabilistic logic

Probabilistic logic provides a formalism that extends propositional logic for dealing with uncertain knowledge [14]. As propositional logic, probabilistic logic also uses propositional variables to represent categorical statements. Propositional variables, or atomic formulas, can assume one of two states, *true* or *false*, and can be combined in order to form a compound formula. A compound formula describes a complex proposition and is obtained by connecting atomic or other compound formulas using the logical operators. In this work, atomic formulas are denoted by lower case letters as $p \ldots, q$, compound formulas are denoted by capital letters $A, B \ldots, C$ and logic operators ($\wedge, \vee$ and $\neg$) has the usual semantics [15]

Let S_i be a formula, atomic or not. Probabilistic logic assumes that the agent's belief in S_i can be represented by a probability assignment $P(S_i) = \pi_i$, with $\pi_i \in [0, 1]$. If beliefs are imprecise, they can be expressed by inequalities as $P(S_i) \geq \pi_i$ or $P(S_i) \leq \pi_i$ or by interval probability statements as $\underline{\pi}_i \leq P(S_i) \leq \overline{\pi}_i$; here $\underline{\pi}_i$ and $\overline{\pi}_i$ are the lower and upper bounds for π_i. Furthermore, exact conditional probability statements expressing the expectation in a sentence S_i given the event S_j can be written as $P(S_i|S_j) = \pi_{i,j}$, $P(S_i|S_j) \geq= \pi_{i,j}$. As before, imprecise conditional beliefs can be represented as inequalities.

A probabilistic logic knowledge base is said consistent if its assignments agree with probability theory axioms. So, if $\mathcal{M}$ denotes all possible truth assignments[2] on the variables of a consistent knowledge base, then:

$$P(S_i) = \sum_{w:\mathcal{M}(S_i,w)} P(w)\,. \tag{2}$$

where $P(w)$ is the probability of a truth assignment.

A *probabilistic logic inference* [16] aims at computing $\underline{P}(S)$ and $\overline{P}(S)$, the lower and upper probability of a sentence S given the constraints defined by a knowledge base with assessments $P(S_*) = \pi_*$, $P(S_*) \leq \pi_*$ $P(S_*) \geq$

[1] In this work we assume a variable can be categorical, discrete or continuous.

[2] A truth assignment is a vector assigning value either true or false to each propositional variable of an expression to the constants *true* or *false*.

π_*. The resultant interval $[\underline{P}(S), \overline{P}(S)]$ must be consistent with the assessments about every sentence S_* and with the probability theory axioms. In this scheme, the given initial assessments compose a knowledge base which encodes the prior information as also any relevant evidence.

Inference can be carried out by linear programming. Basically, let $\mathbf{S} = \{S_1, \ldots, S_m\}$ be a set of propositional sentences associated to a collection of probability assignments $\Pi = \{\pi_1 \ldots \pi_m\}$ and let S be the sentence of interest whose probability is unknown. Let $\mathbf{a}_i$ be a row vector so that the j^{th}element of $\mathbf{a}_i$ is 1 if S_i is true in w_j, the j^{th} truth assignment, otherwise it is zero. Let also $\mathbf{p} = (p_1 \ldots, p_{2^n})^T$ be a vector with the probability of every truth assignment defined on the atomic sentences in $\mathbf{S} \cup \{S\}$. From Equation 2 it is easy to see that $P(S_i) = \mathbf{a}_i^T \mathbf{p}$. A similar vector $\mathbf{a}$ can also be defined for S and the linear programming related to $P(S)$ can be written as:

$$\begin{array}{ll} \min / \max & \mathbf{a}^T \mathbf{p} \\ s.t & A_{m \times 2^n} \mathbf{p} = \pi \\ & p_i \geq 0, \ i = 1..2^n \\ & \mathbf{1p} = 1 \end{array}$$

In this program, $\mathbf{a}_1 \ldots \mathbf{a}_m$ are the rows of matrix A.

III. Probabilistic logic programming and interestingness analysis

Let $F_1 \wedge F_2 \cdots \wedge F_t \rightarrow C$ be a classification rule and S a logical variable which is equivalent to that. Let S_i be a propositional variable that stands for the antecedent F_i. As before, F_i represents a statement in the form $X_j = x_{j,k}$, $X_j \geq x_{j,k}$ or $X_j \leq x_{j,k}$. The consequent C is a sentence S_0 indicating a class labeling.

In the proposed approach, the first step for the subjective interestingness analysis of a rule is to elicit the marginal probabilities of rule antecedents. In this work, it is assumed that this information can be elicited from experts [10], obtained from previous data analysis results [17], learned from domain literature [18] or derived by meta-analysis [11].

Alternatively, the probability of each rule component could be calculated from the empirical or theoretical marginal densities related to its respective attribute [19]. So, let it be an initial assumption that $p(X_j)$ is known for every $X_j \in \mathbf{X}$. In this case, it is always possible to determine the value of π_i for every S_i and then to generate the constraint $P(S_i) = \pi_j$. Given that, lower and upper bounds for $P(S)$ can be calculated by solving the next probabilistic program:

$$\begin{array}{ll} \min / \max & P(S) \\ s.t. & P(S_1) = \pi_1 \\ & \ldots \\ & P(S_t) = \pi_t \end{array} \tag{3}$$

Now, a problem can arise whether $p(X_j)$ is unknown or uncertain. Nevertheless, the proposed approach can be extended in many ways to deal with this issue. For example, a first course of action would be, as stated before, to explore domain experts knowledge or literature. A second choice would be to use ignorance priors [20]. A third alternative would be to employ the imprecise probability theory [21], [3] to obtain a probability interval $[\underline{\pi}_i, \overline{\pi}_i]$ which could be used to define the expression $\underline{\pi}_i \leq P(S_i) \leq \overline{\pi}_i$. In any case, the resultant equations and inequalities could be appended to Program (3) in order to proceed the inferences.

After encoding relevant information in Program (3), it can be converted into a linear program for later solving. Example 1 illustrates the process described here.

> Example 1: Let X_1 and X_2 be two normally distributed variables so that $X_1 \sim N(1; 0.1)$ and $X_2 \sim N(4; 1)$ and let $C = c_1$ be a class assignment whose prevalence is greater or equal to 0.6. Given a rule $S \equiv (X_1 \leq \wedge X_2 \leq 5 \rightarrow C = c_1)$, it is possible to use the previous information to build a probabilistic logic program for $P(S)$. In that program, $P(S_1) = 0.16$, $P(S_2) = 0,84$ and $P(S_0) \geq 0,6$. The upper and lower bounds for $P(S)$ are obtained by solving the next:
>
> $$\begin{array}{ll} \min / \max & \mathbf{ap} \\ s.t & \mathcal{A} \times \mathbf{p} = \Pi \\ & \mathbf{1p} \\ & p_i \geq 0, i = 1..8. \end{array}$$
>
> In this program, $\mathcal{A} = \begin{bmatrix} \mathbf{a}_0 \\ \mathbf{a}_1 \\ \mathbf{a}_2 \end{bmatrix}$, $\mathbf{p} = \begin{pmatrix} p_1 \\ \ldots \\ p_8 \end{pmatrix}$ and $\Pi = \begin{pmatrix} 0.65 \\ 0.16 \\ 0.84 \end{pmatrix}$. The rows of $\mathcal{A}$ and the objective function are defined as $\mathbf{a}_0 = (1,0,1,0,1,0,1,0)$, $\mathbf{a}_1 = (1,1,1,1,0,0,0,0)$, $\mathbf{a}_2 = (1,1,0,0,1,1,0,0)$ and $\mathbf{a} = (1,0,1,1,1,1,1,1)$.

It can also be the case that domain experts do not feel comfortable to assign bounds to the probabilities of some sentences but have information that allows to ascertain comparative probability statements. For example, let Q_1, Q_2 and Q_3 be three sentences defined on $S_1 \ldots, S_t$ so that experts know that: (a) Q_1 is as probable as or more probable than Q_2 and (b) Q_3 is as probable as or more probable than Q_1. This kind of statement can be incorporated to Program (3) as $P(Q_1) \geq P(Q_2)$ and $P(Q_3) \geq P(Q_1)$.

More formally, if it is known that $P(Q_1) \leq P(Q_2)$, $P(Q_1) \geq P(Q_2)$ or $P(Q_1) = P(Q_2)$, it is possible to use that qualitative information to generate expressions, on the form $P(R_1) - P(R_2) \leq 0$, $P(R_1) - P(R_2) = 0$ or $P(R_1) - P(R_2) \geq 0$, respectively. As before, such constraints can be rewritten using a vectorial notation by doing $\mathbf{b}_{1,2} = \mathbf{b}_1 - \mathbf{b}_2 \odot 0$. $\mathbf{b}_1$ and $\mathbf{b}_2$ are the row vectors relative to $P(Q_1)$ and $P(Q_2)$. Given a set of those constraints, grouped into a system $\mathcal{B} \times \mathbf{p} \odot \varpi$, they can be appended to the inference

program as follows:

$$\begin{array}{ll} \min / \max & \mathbf{c}^T \mathbf{p} \\ s.t & \left[\begin{array}{c} \mathcal{A} \\ \mathcal{B} \end{array} \right] \times \mathbf{p} \odot \left[\begin{array}{c} \varpi \\ \mathbf{0} \end{array} \right] \\ & \mathbf{1p} \\ & p_i \geq 0, i = 1..2^t. \end{array} \quad (4)$$

As it can be deduced from the expression above, by encoding the information provided by comparative probabilities into the linear program, it is possible to reduce the solution space of the optimization problem. So, it can contribute to obtaining tighter bounds for $P(S)$ and derived interestingness scores.

A. Dealing with correlation data

Berleant and Jianzhong [22] and Berleant *et al* [23] present a procedure that allows the calculation of envelopes for joint probabilities from Pearson's correlation coefficient and marginal data. This section explores that procedure to draw out additional probabilistic constraints for interestingness analysis.

Initially, let X_i and X_j be two distinct continuous attributes, with known densities $p(X_i)$ and $p(X_j)$, and linear correlation r. Discretization of X_i and X_j into n_1 and n_2 bins introduces two discrete variables, Z and Y whose sample spaces are $\Omega_z = \{z_1 \dots z_{n_1}\}$ and $\Omega_y = \{y_1 \dots y_{n_2}\}$. In addition, let $p(Z)$ and $p(Y)$ be the marginal distributions of these new variables so that their entries are obtained from $p(X_i)$ and $p(X_j)$ by doing $P(z_k) = P\big(\underline{x}_{i,k} < X_i \leq \overline{x}_{i,k}\big)$ and $P(y_l) = P\big(\underline{x}_{j,l} < X_j \leq \overline{x}_{j,l}\big)$. Here $\underline{x}_{i,k}$ and $\overline{x}_{i,k}$ ($\underline{x}_{j,l}$ and $\overline{x}_{j,l}$) are the limits of the k^{th} (l^{th}) bin of Z (Y).

Given $S_i \equiv (X_i \geq x_i)$ and $S_j \equiv (X_j \geq x_j)$, two antecedents of a classification rule, if it is assumed that Z has a value z_a so that $\underline{x}_{i,a} = x_i$ and Y has a value y_b where $\underline{x}_{j,b} = x_j$, $P(S_i)$ and $P(S_j)$ can be easily written in terms of $p(Z)$ and $p(Y)$. Moreover, marginalization of $p(Z,Y)$ produces:

$$\begin{array}{l} P(S_i) = \sum_{k=z_a}^{z_{n_1}} \sum_{l=1}^{n_2} P(Z = z_k \wedge Y = y_l) = \pi_i \\ P(S_j) = \sum_{l=y_b}^{k_{n_2}} \sum_{k=1}^{n_1} P(Z = z_k \wedge Y = y_l) = \pi_j \end{array} \quad (5)$$

Similarly, $P(S_i \wedge S_j)$ can be formulated in terms of $p(Z,Y)$ by doing:

$$P(S_i \wedge S_j) = \sum_{t_* \in \mathbf{t}} P(Z = z_{t_*} \wedge Y = y_{t_*}) = \pi_{i \wedge j}. \quad (6)$$

In this expression, $\pi_{i \wedge j}$ denotes the unknown value $P(S_i \wedge S_j)$ and $\mathbf{t}$ is a vector of pairs of indexes so that, for all $t_* = (k,l) \in \mathbf{t}$, the intervals represented by z_k and y_l agree with the condition symbolized by $S_i \wedge S_j$. As before, Equations 5 and 6 can be represented in a vectorial form and appended to the Program (4).

The point here is that previous equations relate the joint distribution of $p(X_i, Xj)$ to $P(S_i \wedge S_j)$, $P(S_i)$ and $P(S_j)$ - all of them are relevant to compute the probability of the classification rule under analysis. The problem is that integrating that information into described approach depends on estimate or bound the value of $\pi_{i \wedge j}$. It can be done by exploring Equations 7 and 8:

$$\sum_{k,l}^{n_1,n_2} \overline{z_k y_l} P(Z = z_k \wedge Y = y_l) \geq \underline{\mu_i \mu_j} + r \sqrt{\underline{\sigma_i^2 \sigma_j^2}} \quad (7)$$

$$\sum_{k,l}^{n_1,n_2} \underline{z_k y_l} P(Z = z_k \wedge Y = y_l) \leq \overline{\mu_i \mu_j} + r \sqrt{\overline{\sigma_i^2 \sigma_j^2}} \quad (8)$$

As shown by Berleant and Jianzhong [22] those equations can be used to calculate an outer envelope for $p(Z,Y)$ given correlation data and some statistical measures. Basically, beyond the correlation of X_i and X_j, their utilization requires that outer bounds on the expected values ($\underline{\mu_*}$ and $\overline{\mu_*}$) and variances ($\underline{\sigma_*^2}$ and $\overline{\sigma_*^2}$) be known.

Equations 5, 6, 7 and 8 can be grouped to form a linear system $\mathcal{D}$ that also stores the constraints implied by the probability theory. Appending $\mathcal{D}$ to Program (4) can be useful for two reasons. Firstly, because it provides additional constraints to the optimization program and, from this, contributes to obtaining tighter intervals for $P(S)$. Secondly, because it also allows the collection of information in another way.

The last point to be discussed here is a note about the acquisition of $\underline{\mu_j}$, $\overline{\mu_j}$, $\underline{\sigma_i^2}$, $\underline{\sigma_j^2}$, $\overline{\sigma_i^2}$, $\overline{\sigma_j^2}$. As proposed by Berleant and Jianzhong (2004) and Berleant *et al* (2007), this work assumes that those limits are entered by the analyst or calculated by interval optimization upon $P(Z)$ and $P(Y)$.

B. Evaluating interestingness

The described approach assumes that interestingness analysis is performed as a post-processing routine. That is, interestingness analysis is performed after the data mining step and it aims at sorting or filtering the mined rules according to their interestingness scores.

Given that, calculating the level of interestingness of a rule starts by computing $P(S)$ with the model previously described and then employing that result to determine the numerator, $c(S) = -\log(P(S)) - \log(P(\mathbf{D}|S))$ in Equation 1. If $P(S)$ is determined exactly, $c(S)$ can be calculated directly by using Equation 1. Otherwise, if the result is an interval $[\underline{P}(S), \overline{P}(S)]$, Equation 1 can be used to derive an interval for $c(S)$. In this case, since $\underline{c}(S) = -log(\overline{P}(S)) - log(P(\mathbf{D}|S))$ and $\overline{c}(S) = -log(\underline{P}(S)) - log(P(\mathbf{D}|S))$ are the limits of such interval, the minimum and maximum of level score are given by :

$$\begin{array}{l} \underline{level}(S) = 1 - \frac{\overline{c}(S)}{c(S_0)} \\ \overline{level}(S) = 1 - \frac{\underline{c}(S)}{c(S_0)} \end{array} \quad (9)$$

After obtaining the interval for $level(S)$, the interestingness analysis continues by inspecting its lower and upper bounds. if $\underline{level}(S)$ is greater than 0, it means that the rule appears to be interesting given prior knowledge as also effective in describing data, even if it was computed on the lower bound for $P(S)$. On the other hand, $\overline{level}(S) < 0$ is

an indicative that, in the light of background information, the rule is not interesting even if the analysis considers an upper bound for $P(S)$. Finally, if $0 \in [\underline{level}(S), \overline{level}(S)]$ no direct conclusion can be drawn.

IV. An application example

This section shows an application that uses the proposed approach to calculate the level of interestingness of classification rules induced by JRIP algorithm [24], [25] for the Breast Cancer Wisconsin Data Set [26]. The 569 cases in data set were split into two partitions, the training data with $2/3$ of the instances and a test data set with the rest. The JRIP algorithm generated the following rules:

- rule (a): (concave points n1 $\geq$ 0.05182) and (perimeter n3 $\geq$ 113.9) $\rightarrow$ Diagnosis=malign;
- rule (b): (concave points n1 $\geq$ 0.05839) and (texture n3 $\geq$ 23.75) $\rightarrow$ Diagnosis=malign;
- rule (c): (radius n3 $\geq$ 15.65) and (texture n3 $\geq$ 28.06) and (smoothness n3 $\geq$ 0.1094) $\rightarrow$ Diagnosis=malign.

The attributes on the left side of these rules refer to some features of cellular nucleous. The right side indicates positive diagnostic of malignancy.

As prescribed by the proposed approach, the first rule was associated with a sentence $S_a \equiv S_1 \wedge S_2 \rightarrow S_0$ where S_1 and S_2 symbolize the conditions *concave points n1* $\geq$ *0.05182* and *perimeter n3* $\geq$ *113.9*, respectively. S_0 denotes the sentence *Diagnosis=malign*. The prior probabilities of S_1 and S_2 were set as $P(S_1) = \pi_1 = 0.41$ and $P(S_2) = \pi_2 = 0.36$ and the conditional probability $P(S_1|S_0)$ was also entered as $\pi_{1,0} = 0.82$. In the example it was assumed that those values were informed by an hypothetical expert[3]. The prevalence of breast cancer incidence[4] in US was defined as $P(S_0) = \pi_0 = 0.001$. Next, the following probabilistic logic program was written as:

$$\begin{array}{ll} \min / \max & P(S_a) \\ s.t & P(S_0) = 0.001 \\ & P(S_1) = 0.41 \\ & P(S_2) = 0.36 \\ & P(S_1|S_0) = 0.82 \end{array}$$

Similarly, Rule (b) was associated with a sentence rule $S_b \equiv S_3 \wedge S_4 \rightarrow S_0$ where S_3 and S_4 represent the propositions *concave points n1* $\geq$ *0.05839* and S_4 and *texture n3* $\geq$ *23.75*. The input probabilities were fixed as $P(S_3) = 0.34$, $P(S_4) = 0.58$ and $P(S_4|S_0) = 0.85$. Rule (c) was related to sentence $S_c \equiv S_5 \wedge S_6 \wedge S_7 \rightarrow S_0$ where S_5, S_6 and S_7 indicate *radius n3* $\geq$ *15.65*, *texture n3* $\geq$ *28.06* and *smoothness n3* $\geq$ *0.1094*, respectively. The input probabilities were $\pi_5 = 0.4987$, $\pi_6 = 0.3398$, $\pi_7 = 0.8452$ and $\pi_{6,0} = 0.4425$.

After that, the revised simplex algorithm was used to calculate intervals for $P(S_a)$, $P(S_b)$ and $P(S_c)$, the obtained results were $[0.64, 1]$, $[0.66, 1]$ and $[0.66, 1]$, respectively. Next, the calculated probability intervals were combined with the likelihoods (see [5]) of S_a, S_b and S_c in order to calculate lower and upper bounds for the level score. The results were $level(S_a) \in [0.65, 0.75]$, $level(S_b) \in [0.026, 0.026]$ and $level(S_c) \in [0.023, 0.023]$. These results indicate that only the first rule seems to have a considerable degree of interest when confronting data and background knowledge.

In the sequence, it is supposed that the analyst has two additional pieces of information he wants to take into account when evaluating the first rule. The first one informs a qualitative constraint $P(S_1 \wedge S_0) \geq P(S_2 \wedge S_0)$. The second one, supplied by an expert, declares that the expected value of *concave points n1* and *perimeter n3* are bounded by the intervals $[75.22, 138.8]$ and $[0.05; 0.08]$ while their variances pertain to intervals $[28, 34]$ and $[0.032, 0.044]$.

By solving the linear program updated with this information, a new belief interval for S_a is $P(S_a) \in [0.99, 1]$. It makes that the lower bound for $level(S_a)$ be updated to 0.66.

V. Discussion

The probabilistic approach presented in this work provides a basic scheme to encode into a knowledge base information acquired from experts, literature or statistical reports aiming to make advances in the interestingness analysis of classification rules. In particular, it allows using information elicited from experts or descriptive statistical data to assess the marginal or joint probabilities for the propositions which compose a classification rule. Once this kind of information is often available in several domains [11] [22], the proposed approach can be useful in many situations.

Additionally, by exploring the probabilistic logic language, the presented approach implements two facilities which are inherent to that logic. A first facility stems from the fact that probabilistic logic reasoning is able to deal with uncertain and incomplete knowledge [27], [28], [29], qualitative probabilities [30] and imprecise beliefs [31] using the same inference engine - linear programming. A second one is that probabilistic logic modeling does not require the construction of a complete probabilistic model in order the encoding sentences involving many domain variables. This way, since the statements in the knowledge base comply with the axioms of probability theory [32], the declaration of any statement (with few or several terms) does not depend on prior statements.

Many other authors have used probabilistic reasoning methods in subjective interestingness analysis [33] [34][35] [36]. In particular, the present proposal bears some similarities with those described by Jaroszewicz and Simovici [37] and Malhas [8]. Those authors show schemes for subjective interestingness analysis of association rules where they apply the formalism of Bayesian networks [38] to represent the domain knowledge and proceed the computation of an interestingness score. That design choice allows them to take advantage from the expressiveness of the language and efficiency of reasoning algorithms provided by the Bayesian network formalism. However, Bayesian networks

[3]For practical reasons, $P(S_1)$, $P(S_2)$ and $P(S_1|S_0)$ were estimated from a random sample extracted from the original data set.

[4]See http://www.cdc.gov/mmwr/preview/mmwrhtml/00043942.htm.

reasoning assumes that model probabilities are precise and the treatment of imprecise probabilities often demands the employing of extended formalisms [39] [40] [41]. Give that, the present approach can be viewed as an alternative for domains which are well represented by rule-based systems and the development team does not have time or resources to specify a complete probabilistic model.

Finally, it must be noted that: (a) probabilistic logic inference is a time-consuming task [16]; (b) depending on the application and the learning strategy, rule miners can generate too many patterns [42]; and (c) it is desirable that interestingness analysis tools have a pleasing time performance. So, given (a) and (b), it could be argued that the presented scheme would be ineffective to achieve (c) if it is necessary to reason on a very large knowledge base or if there are too many rule patterns to process. However, a further scrutiny shows that rule mining algorithms, usually, implement a kind of Occam's razor or strategy for rule pruning. So, for some applications, the mined rules will not have many components (propositions). In such cases, it is expected that the inference problem does not have too many elements to process and, therefore, it can be solved quickly. Additionally, empirical results described in Cozman, de Campos and da Rocha [43], Jaumard, Hansen and Aragão [14] and Hansen *et al* [31], shows that the use of column generation technique [44] allows to solve the linear programs related to probabilistic logic programs with a few dozens of variables and two hundred of sentences in less than a minute (few seconds). This time performance can be acceptable for a number of real world tasks.

VI. Conclusion

This work presented a propositional probabilistic logic-based approach for subjective interestingness analysis of classification rules. An advantage of the proposed approach is that it allows to integrate domain knowledge from experts, literature and statistical reports to compute probability based-interestingness measures. Another advantage is that it is possible to carry out valid probabilistic reasoning even if available knowledge is uncertain or incomplete and elicited beliefs are imprecise.

As a future work it is intended to extend the proposed approach to integrate independence assumptions into the inference step by combining probabilistic logics and graph based representations (see [45] and [43]). Another objective is to investigate the possibility of using the proposed scheme to validate or review prior beliefs, given the mined rules (similar to the soft belief analysis described by Silberschatz and Tuzhilin [36]). Finally, the development of an extended approach for dealing with interestingness analysis of association rules is also intended.

Acknowledgment

Thanks to CAPES, CNPq, Finep and Fundação Araucária for partially support this work.

References

[1] S. Chen and B. Liu, "Generating classification rules according to user's existing knowledge," in *First SIAM International Conference on Data Mining*, 2011, pp. 1–15.

[2] B. Liu, W. Hsu, and S. Chen, "Using general impressions to analyze discovered classification rules," in *3rd Intl. Conf. on Knowledge Discovery and Data Mining*, 1997, pp. 31–36.

[3] P. Walley, *Statistical Reasoning with Imprecise Probabilities*, ser. Monographs on Statistics and Applied Probability. London: Chapman and Hall, 1991.

[4] ——, "Measures of uncertainty in expert systems," *Artificial Intelligence*, vol. 83, no. 1, pp. 1–58, 1996.

[5] D. Gay and M. Boullé, "A bayesian criterion for evaluating the robustness of classification rules in binary data sets," in *Advances in Knowledge Discovery and Management*, 2013, pp. 3–21.

[6] J. Vashishtha, D. Kumar, S. Ratnoo, and K. Kundu, "Article: Mining comprehensible and interesting rules: A genetic algorithm approach," *International Journal of Computer Applications*, vol. 31, no. 1, pp. 39–47, October 2011.

[7] U. Fayyad, G. Piatestky-Shapiro, and P. Smyth, "From data mining to knowledge discovery in databases," *AI Magazine*, vol. 17, no. 3, pp. 37–54, 1996.

[8] R. Malhas and Z. A. Aghbari, "Interestingness filtering engine: Mining bayesian networks for interesting patterns," *Expert Syst. Appl.*, vol. 36, no. 3, pp. 5137–5145, 2009.

[9] T. Yu, S. Simoff, and D. Stokes, "Incorporating prior domain knowledge into a kernel based feature selection algorithm," in *Advances in Knowledge Discovery and Data Mining*, ser. Lecture Notes in Computer Science, Z.-H. Zhou, H. Li, and Q. Yang, Eds. Springer Berlin Heidelberg, 2007, vol. 4426, pp. 1064–1071.

[10] A. O'Hagan, C. Buck, A. Daneshkhah, J. Eiser, P. Garthwaite, D. Jenkinson, J. Oakley, and T. Rakow, *Uncertain judgements: eliciting experts' probabilities*. Chichester: John Wiley and Sons, 2006.

[11] Y. Barbaros, Z. Perkins, T. Rasmussen, N. Tai, and D. Marsh, "Combining data and meta-analysis to build bayesian networks for clinical decision support," *Journal of Biomedical Informatics*, p. in press, 2014.

[12] A. A. Silva and F. de Souza, "A protocol for the elicitation of imprecise probabilities." in *4th Intl. Symp. on Imprecise Probabilities: theory and applications*, F. Cozman, R. Nau, and T. Seidenfeld, Eds. SIPTA, 2005, pp. 315–321.

[13] L. G. de O. Silva and A. A. Filho, "A method for elicitation and combination of imprecise probabilities: A mathematical programming approach," in *2014 IEEE International Conference on Systems, Man, and Cybernetics*, 2014, pp. 619–624.

[14] B. Jaumard, P. Hansen, and M. D. Aragao, "Column Generation Methods for Probabilistic Logic," in *Integer Programming and Combinatorial Optimization*, 1990, pp. 313–331.

[15] A. Hamilton, *Logic for Mathematicians*. Cambridge University Press, 1988.

[16] P. Hansen and B. Jaumard, "Probabilistic satisfiability," École Polytechique de Montréal, Technical Report G-96-31, 1996.

[17] P. H. Garthwaite, J. B. Kadane, and A. O'Hagan, "Statistical methods for eliciting probability distributions," *Journal of the American Statistical Association*, vol. 100, pp. 680–701, 2005.

[18] L. van der Gaag, S. Renooij, C. Witteman, B. Aleman, and B. Taal, "How to elicit many probabilities," in *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*, Stockholm, Sweden, 1999, pp. 647–654.

[19] D. S. Sivia, *Data Analysis, A Bayesian Tutorial*. New York: Oxford University Press, 1996.

[20] P. Gregory, *Bayesian Logical Data Analysis for the Physical Sciences*. New York, NY, USA: Cambridge University Press, 2005.

[21] I. Levi, *The Enterprise of Knowledge*. Cambridge: MIT Press, 1980.

[22] D. Berleant and Z. Jianzhong, "Using pearson correlation to improve envelopes around the distributions of functions." *Reliable Computing*, vol. 10, no. 2, pp. 139–161, 2004.

[23] D. Berleant, M. Ceberio, G. Xiang, and V. Kreinovich, "Towards adding probabilities and correlations to interval computations," *Int. J. Approx. Reasoning*, vol. 46, no. 3, pp. 499–510, 2007.

[24] W. W. Cohen, "Fast effective rule induction," in *In Proceedings of the Twelfth International Conference on Machine Learning*. Morgan Kaufmann, 1995, pp. 115–123.

[25] B. Martin, "Instance-based learning : Nearest neighbor with generalization," Tech. Rep., 1995.

[26] W. Wolberg., W. Street, and O. Mangasarian, "Machine learning techniques to diagnose breast cancer from image-processed nuclear features of fine needle aspirates." *Cancer Lett*, vol. 77, no. 2-3, pp. 163–71, 1994. [Online]. Available: http://www.biomedsearch.com/nih/Machine-learning-techniques-to-diagnose/8168063.html
[27] K. A. Andersen and J. N. Hooker, "Bayesian logic," *Decision Support Systems*, vol. 11, no. 2, pp. 191–210, Feb 1994.
[28] R. Haenni, J.-W. Romeijn, G. Wheeler, and J. Williamson, *Probabilistic Logics and Probabilistic Networks*. Springer Publishing Company, Incorporated, 2013.
[29] J. N. Hooker, "Mathematical programming methods for reasoning under uncertainty," in *Operations Research 1991*, ser. Operations Research Proceedings 1991, W. Gaul, A. Bachem, W. Habenicht, W. Runge, and W. Stahl, Eds., vol. 1991. Springer Berlin Heidelberg, 1992, pp. 23–34.
[30] Z. Ognjanovi, A. Perovi, and M. R. kovi, "An axiomatizatin of qualitative probability," *Acta Polytechnica Hungarica*, vol. 1, no. 5, pp. 105–110, 2008.
[31] P. Hansen, B. Jaumard, M. P. de Aragão, F. Chauny, and S. Perron, "Probabilistic satisfiability with imprecise probabilities," *International Journal of Approximate Reasoning*, vol. 24, no. 23, pp. 171 – 189, 2000.
[32] P.S.S.Andrade, J. da Rocha, D.P.Couto, A.C.Teves, and F.G.Cozman, "A toolset for propositional probabilistic logic," in *Encontro Nacional de Inteligłncia Artificial*. SBC, 2007, pp. 1371–1380.
[33] K. McGarry, "A survey of interestingness measures for knowledge discovery," *Knowl. Eng. Rev.*, vol. 20, no. 1, pp. 39–61, 2005.
[34] T. D. Bie, "Maximum entropy models and subjective interestingness: an application to tiles in binary databases." *Data Mining and Knowledge Discovery*, vol. 23, no. 3, pp. 407–446, 2011.
[35] G. Dong and J. Li, "Interestingness of discovered association rules in terms of neighborhood-based unexpectedness," in *Research and Development in Knowledge Discovery and Data Mining*, ser. Lecture Notes in Computer Science, X. Wu, R. Kotagiri, and K. Korb, Eds. Springer Berlin Heidelberg, 1998, vol. 1394, pp. 72–86.
[36] A. Silberschatz and A. Tuzhilin, "What makes patterns interesting in knowledge discovery systems," *IEEE Transactions on Knowledge and Data Engineering*, vol. 8, no. 6, pp. 970–974, 1996.
[37] S. Jaroszewicz, T. Scheffer, and D. A. Simovici, "Scalable pattern mining with bayesian networks as background knowledge." *Data Min. Knowl. Discov.*, vol. 18, no. 1, pp. 56–100, 2009.
[38] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco: Morgan Kaufmann, 1988.
[39] F. Cozman, "Graphical models for imprecise probabilities," *Int. J. Approx. Reasoning*, vol. 39, no. 2-3, pp. 167–184, 2005.
[40] B. Tessem, "Interval probability propagation," *International Journal of Approximate Reasoning*, no. 7, pp. 95–120, 1992.
[41] M. P. Wellman, "Fundamental concepts in qualitative probabilistic networks," *Artificial Intelligence*, vol. 44, no. 3, pp. 257–303, 1990.
[42] B. V. Balaji and V. V. Rao, "Improved classification based association rule mining," *Intl. Journal of Advanced Research in Computer and Communication Engineering*, vol. 2, no. 5, pp. 2211–2221, May 2013.
[43] F. G. Cozman, C. de Campos, and J. da Rocha, "Probabilistic logic with strong independence," in *Advances in Artificial Intelligence, IBERAMIA-SBIA 2006, Brazilian Symposium on Artificial Intelligence*, ser. Lecture Notes in Computer Science, J. Sichman, H. Coelho, and S. Rezende, Eds., vol. 4140, 2006, pp. 612–621.
[44] J. Desrosiers and M. Lubbecke, *Column Generation*. Boston, USA: Springer, 2005, ch. A Primer in Column Generation, pp. 1–32.
[45] L. C. v. d. Gaag, "Computing probability intervals under independency constraints," in *Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence*. New York, NY, USA: Elsevier Science Inc., 1991, pp. 457–466.

A Hybrid Weighted Nearest Neighbor Approach to Mine Imbalanced Data

Harshita Patel[1], G.S. Thakur[2]
[1,2]Department of Computer Applications,
Maulana Azad National Institute of Technology, Bhopal, Madhya Pradesh, India, 462003

Abstract -*Classification of imbalanced data has drawn significant attention from research community in last decade. As the distribution of data into various classes affects the performances of traditional classifiers, the imbalanced data needs special treatment. Modification in learning approaches is one of the solutions to deal with such cases. In this paper a hybrid nearest neighbor learning approach is proposed for mining binary imbalanced datasets. This hybridization is based on different K for different classes as large K for large classes and small K for small classes and with weighted concept as small weights for large classes and large weight for small classes. The merger of dynamic K improves the performance of weighted approach.*

Keywords: Nearest neighbors, imbalanced data, classification, weights.

1 Introduction

The well developed era of science and technology facilitated the world in huge manner, also results in production of immense amount of data. This ever increasing data need proper treatment for use. Data mining, data science and machine learning are some name of research solutions to treat such data. With every new day many challenges are being solved with various new approaches; learning from imbalanced data is one of them and become a part of interest of data researchers from more than a decade. In imbalanced datasets classes are unequally distributes as some classes overwhelmed by others in terms of number of instances [5][9]. Some examples of imbalanced datasets are Credit Card Fraud Detection [12], Oil-spill Detection [7], Medical Diagnosis [13], Cultural modeling [19], Text Categorization, Network Intrusion, Helicopter Gearbox Fault Monitoring [5] etc.

Previous researches has shown that the presence of imbalance in data causes decreased and many times inaccurate performances of traditional classifiers such as Decision Trees, SVM, Naïve-Bayes and Nearest Neighbors which results in biased classification and may be severe in many cases like for medical data. Four common ways to deal with the problems are (i) balance data by using sampling techniques; i.e. under sampling or over sampling, (ii) modification in traditional classification algorithms, (iii) cost sensitive approaches and (iv) ensemble techniques. In this paper second solution of algorithm modification is adopted for nearest neighbor classification in terms of different *K* value for different classes in weighted nearest neighbor algorithms for imbalanced datasets.

K nearest neighbor algorithm comes under top 10 data mining algorithms [20] due to its simplicity, easy programming, implementation, comprehensiveness and robustness. Its error rate is bounded above by twice the Bayes error rate [16][3]. Like other traditional classifiers, K nearest neighbor also suffers from less accurate and biased performance issue in presence of imbalance in datasets. There are many alternative nearest neighbor approaches have been developed for such cases. Some approaches were proposed for text classification or categorization. Average of Similarity-*K*NN (AS-*K*NN) is proposed by Yang et. al [21], in this approach the sum of similarity of each class is divided by the total number of instances in the nearest neighbor. One algorithm Nearest-Weighted *K*NN (NW-*K*NN) is proposed by Tan [6] in which small size classes are assigned with larger weights and large classes assigned with small weights. Adaptive *K*NN (ADPT-*K*NN) is proposed by Baoli et. al [14] performs on different values of *K* for each class on varying number of training data. The value of *K* is large for larger classes and small for smaller classes. Another method is Drag-Pushing KNN (DP-*K*NN) [15] in which weights of features of classes are increased or decreased in dealing with misclassified data.

Also a lot of work has been done on other real world problems and generalized for any dataset. Numerous modifications on *K*NN have been proposed with weight concepts too. A fine background was provided by algorithms like called WDNN (Weighted Distance Nearest Neighbor) [8] and WDkNN [17] for weighted nearest neighbor approach. WDNN preserves important data points by assigning positive weight values. While WDNN works with on k=1, WDkNN is a next improved step which works on values greater than 1 of k. Chawla et. al. have been shown efficient working of CCW (Class Confidence

Weights) [18] as it considering attribute probability of given class. Dubey et al. [4] proposed another weighted *K*-nearest neighbor algorithm. They considered class distribution for neighbors of any test instance. Initial classification taken from traditional *K*-nearest neighbor algorithm that how it classified an instance and used to calculate the weight for each class. Kriminger et. al. [2] proposed a single class algorithm to decrease the effect of imbalance using the local geometric structure in data. This algorithm applies to diverse degrees of inequality and in addition able to work on any number of classes. It also allows adding new examples to training data sets. Chen et. al. [10] have concentrated on the impact of various measures cost ratio, imbalance ratio and sample size on classification results of a French bankruptcy database and found that such measures have severe impact on the classification performance. Tomasev et. al. [11] discussed about the hubness effect related to *K*-nearest neighbors in high-dimensional datasets that result in high misclassification rate due to minority class examples unlike in small and medium dimensional datasets where misclassification occur due to majority class examples. Ryu et. al. [1] proposed an instance hybrid selection using nearest neighbor (HISNN) for cross-project defect prediction (CPDP) where class imbalance exists in distributions of source and target projects. In this algorithm, local information is learned by *K*-nearest neighbor algorithm while naive Bayes is applied to gain global information. This hybridization results in high performance in software defect prediction.

In this paper a hybrid *K* nearest neighbor is proposed for binary classification by combining the ideas of dynamic *K* for different classes with large and small weights for small and large classes. The structure of this paper contains basic learning in preliminaries in section II, followed by proposed algorithm in section III. Section IV contains experiments and result discussions and paper is concluded in section V.

2 Preliminaries

2.1 *K*-Nearest Neighbor

The *K*-nearest neighbor algorithm finds class label for any instance *q* from test dataset, by finding *K* (some integer) nearest neighbors of *q* from training dataset and then assign the class label for which *q* will have maximum neighbors. It could be shown by following equation:

$$C(q) = \arg\max_{C \in \{Cj \mid j=1,2\}} \sum_{y_i \in S(q,K)} T(y_i, C)$$

Here $C(q)$ = *class label of q, to be predicted,*
m = *Number of classes,*
$S(q,K)$ = *Set of* $K-$*nearest neighbors of q and*

$$T(y_i, C) = \begin{cases} 1 & \text{if } y_i \in C \\ 0 & \text{otherwise} \end{cases}$$

2.2 Adaptive *K*-Nearest Neighbor

Baoli et. al. (2004) [6] proposed the Adaptive *K* nearest neighbor for dealing with imbalanced text corpus. The main idea behind this research is to define different *K* for different classes. The *K* is finding out according to sizes of classes. Altering *K* with classes is giving more accurate categories rather than static *K* for all classes; because uneven distribution of data into classes affects the classification performance.

Selection of K_{Cj} (for particular class) for classes is done using following equation

$$K_{Cj} = \min\left(\lambda + \left\lceil \frac{K * I(C_j)}{\max\{I(C_j) \mid j=1,2\}} \right\rceil, K, I(C_j)\right)$$

Here K = *Original input integer to define nearest neighbors,*
K_{Cj} = *Calculated K for each class* C *using above formula,*
$I(C_j)$ = *Number of instances in class* C_j *where* $j = 1$ *and* 2,
λ = *Constant Integer value.*

2.3 Neighbor Weighted *K*-Nearest Neighbor

Neighbor Weighted K nearest Neighbor approach was proposed by Tan [14] for imbalanced text datasets. The basic concept used in this method is to assign large weight to small classes and small weights to large classes to reduce the biasness of the classifier towards majority class and ignorance of minority one. Weights defined through this method help to improve the performance of nearest neighbor classifier in presence of imbalance in datasets.

First we find *K* nearest neighbors for query instance q, find weight from following equation:

$$W_i = \frac{1}{(N(C_j) / Min\{N(C_j) \mid j=1,2\})^{1/p}}$$

p is an exponent and *p*>1,

And then weight is applied to traditional classification algorithm.

$$C(q) = \arg\max_{C \in \{Cj \mid j=1,2\}} W_i \left(\sum_{y_i \in S(q,K)} T(y_i, C) \right)$$

3 Proposed Method

The proposed nearest neighbor approach merges the concepts of weights and specific K of each class in any binary imbalance data space where one class is overwhelmed with the other one. The parameter K plays an important role in the performance of the nearest neighbor classifier. In presence of imbalance in data space it become more vital as one class dominant the other with more instances. Sometimes minority class contain crucial information as happened in many real world applications, but due to having less or very few representative instances, classical methods unable to classify them correctly. In K nearest neighbor common K for all classes biases the result towards the majority class. Because in case of common K for all classes may implies large number of nearest neighbors consideration for classification for majority and minority both and due to less quantity, minority class instances will be classified as majority class instances. Adaptive approach [6] suggests the dynamic K, with the size of classes. It improves the accuracy of classification for minority class too.

Neighbor weighted method is another solution for imbalanced datasets. This method suggests assigning weights according to size of classes, large weights for small classes and small weights for large classes. Both approaches were proposed for imbalanced text corpus to categorize the imbalance text more accurately. These methods perform well with numeric data too. Combining the both methods give better performance by applying varying K on neighbor weight approach.

Algorithm

Input: Training Dataset D, Query instance q, Set of class labels C and Parameter K.

Output: Class label of q.

Step 1. Find K nearest Neighbor for q

Step 2. Obtain weights with

$$W_x = \frac{1}{(N(C_x) / Min\{N(C_j) \mid j = 1,2\})^{1/p}}$$

Step 3. Find K_{C_j} for classes by using equation

$$K_{C_j} = \min\left(\lambda + \left\lceil \frac{K * I(C_j)}{\max\{I(C_j) \mid j = 1,2\}} \right\rceil, K, I(C_j) \right)$$

Step 4. Determine class label of q with

$$C(q) = \underset{C \in \{Cj \mid j=1,2\}}{\arg\max} \; W_x \Big(\sum_{y_i \in S(q, K_{Cj})} T(y_i, C) \Big)$$

4 Experiments and Results

4.1 Datasets

The experiments with proposed approach are performed for binary classification. Seven imbalanced datasets have been taken with different imbalance ratio. All datasets are processed with their all features and effect of individual feature on classification is not considered so assuming that all features are essential; no feature selection is applied here. Short description of datasets used is given in following table:

TABLE1
DESCRIPTION OF IMBALANCED DATASETS

Datasets	Source	# Instances	Class (1/0)	#Attributes	IR
Phoneme	KEEL	5404	Oral/Nasal Sound	5	2.4
Transfusion	UCI	748	Blood Doneted Yes/No	4	3.2
Vehicle0	KEEL	846	Positive/ Negaive	18	3.3
Yeast-2_vs_4	KEEL	514	Positive/ Negaive	8	9.1
Glass2	KEEL	214	Positive /Negaive <=4/>4	9	11.6
Wine Quality	UCI	4898	Wine Quality >20/<=20	11	25.8
Abalone	UCI	4177	rings	7	66.4

4.2 Evaluation Measures

Evaluating accuracy is not enough for imbalanced datasets. It is possible that a classifier may achieve good accuracy results but accuracy rate would be higher for majority class and very less for minority class due to less number of instances present in dataset. Some known and trendy evaluation measures for classifiers of imbalanced data are F-measure, AUC, G-Mean etc.

The classification results could be seen in terms of confusion metrics:

		Actual Values	
Predicted Values		TP (True Positive)	FP (False Positive)
		FN (False Negative)	TN (True Negative)

Fig 1. Confusion metrics for performance evaluation

TP is the value of actual positives which are correctly classified as positives. FP is the value of actual negatives which are incorrectly classified as positives. Similarly TN is the value of actual negatives that are correctly classified as negatives. FN is the value of actual positives that were incorrectly classified as negatives. Accuracy measures could be shown in terms of these metrics values:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

$$\Pr esicion = \frac{TP}{TP + FP}$$

$$\mathrm{Re}call = \frac{TP}{TP + FN}$$

$$F - Measure = \frac{2 * \Pr ecision * \mathrm{Re}call}{\Pr ecision + \mathrm{Re}call}$$

F-measure is a popular evaluation measure to evaluate classification performance for imbalanced data and for other machine learning tasks. Table 2 describes the F-measure values of neighbor weighted approach and our hybrid method:

TABLE II
F-MEASURE FOR DIFFERENT VALUES OF *K*

Dataset	K	NWKNN	Hybrid Approach
Phoneme	5	0.4548	0.4548
	10	0.4551	0.4562
	15	0.454	0.4549
	20	0.4497	0.4551
	25	0.4503	0.4568
Transfusion	5	0.3118	0.375
	10	0.2429	0.3523
	15	0.1923	0.3978
	20	0.225	0.4277
	25	0.0606	0.4314
Vehicle0	5	0.3491	0.381
	10	0.3698	0.4096
	15	0.3465	0.4
	20	0.3629	0.4152
	25	0.339	0.4138
Yeast-2_vs_4	5	0.1429	0.1529
	10	0.1606	0.1678
	15	0.1626	0.2029
	20	0.1897	0.1986
	25	0.1622	0.2
Glass2	5	0.1	0.0938
	10	0.0408	0.1449
	15	0.0465	0.127
	20	0.0541	0.1667
	25	0.0606	0.1449
Wine Quality	5	0.0526	0.0636
	10	0.0466	0.0721
	15	0.0432	0.0721
	20	0.0418	0.0721
	25	0.0383	0.0721

The table shows the improvement in performance in terms of F-measure. Table III shows the AUC for neighbor weighted and our method; though it is still very less but significantly improved with the neighbor weighted approach.

TABLE III
AUC FOR DIFFERENT VALUES OF *K*

Dataset	K	NWKNN	Hybrid Approach
Phoneme	5	0.5017	0.5017
	10	0.5031	0.5044
	15	0.5022	0.5027
	20	0.4963	0.5031
	25	0.4988	0.5069
Transfusion	5	0.4677	0.5168
	10	0.4545	0.509
	15	0.4738	0.5667
	20	0.532	0.6049
	25	0.4865	0.6004
Vehicle0	5	0.4696	0.5079
	10	0.5063	0.5541
	15	0.4801	0.5381
	20	0.5064	0.5644
	25	0.4828	0.5619
Yeast-2_vs_4	5	0.4062	0.43235
	10	0.46735	0.48275
	15	0.48085	0.57455
	20	0.5429	0.56375
	25	0.48705	0.56735
Glass2	5	0.3593	0.3254
	10	0.2356	0.5
	15	0.28645	0.44235
	20	0.3373	0.57625
	25	0.3712	0.5
Wine Quality	5	0.392	0.4512
	10	0.3821	0.5
	15	0.3883	0.5
	20	0.4002	0.5
	25	0.4026	0.5

5 Conclusions and Future Work

In this paper adaptive *K* is merged with neighbor weighted approach and it helps to improve the classification performance of neighbor weighted approach. Neighbor weighted approach was in itself classifying well the

imbalanced data. But with the use of small K for small class and large K for large class, classification results get improved. The research was done with two class dataset and it could be further performed with multiclass datasets. AUC is better than the neighbor weighted approach and needs more improvement. Enhancement of the proposed algorithm could be more refined with better performances.

6 References

[1] D. Ryu, J. Jang and J. Baik "A hybrid instance selection using nearest-neighbor for cross-project defect prediction", *Journal of Computer Science and Technology*, vol. 30, no. 5, pp. 969-980, 2015.

[2] E. Kriminger and C. Lakshminarayan, "Nearest neighbor distributions for imbalanced classification", *In Proc. of WCCI 2012 IEEE World Congress on Computational Intelligence*, Brisbane, pp. 10-15, 2012.

[3] G. Loizou and S. J. Maybank. "The nearest neighbor and the bayes error rates", *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 9, no. 2, pp. 254-262, 1987.

[4] H. Dubey and V. Pudi, " Class based weighted k nearest neighbor over imbalanced dataset" *PAKDD 2013, Part II, LANI 7819*, pp. 305-316, 2013.

[5] H. He and E. A. Garcia, "Learning from Imbalanced Data", *IEEE Transactions on Knowledge and Data Engineering*, vol.21, no.9, pp.1263-1284, 2009.

[6] L. Baoli, L. Qin and Y. Shiwen, "An adaptive k nearest neighbour text categorization strategy", *ACM Transactions on Asian Language Information Processing*, vol. 3, no. 4, pp. 215-226, 2004.

[7] M. Kubat, R. C. Holte and S. Matwin, "Machine Learning for the Detection of Oil spills n Satellite images", *Machine Learning*, vol. 30, nos. 2/3, pp. 195–215, 1998.

[8] M. Z. Jahromi, E. Parvinnia and R. John, "A method of learning weighted similarity function to improve the performance of nearest neighbor", *Information Sciences*, vol. 179, pp. 2964–2973, 2009.

[9] N. V. Chawla, *Data Mining for Imbalanced Datasets: An overview*, Data Mining and Knowledge Discovery Handbook, pp.853-867.

[10] N. Chen, A. Chen and B. Ribeiro "Influence of class distribution on cost-sensitive learning: A case study of bankruptcy analysis", *Intelligent Data Analysis*, vol. 17, no. 3, pp. 423-437, 2013.

[11] N. Tomašev and D. Mladenic, "Class imbalance and the curse of minority hubs", *Knowledge-Based Systems*, vol. 53 pp. 157–172, 2013.

[12] P. Chan and S. Stolfo, "Toward scalable learning with non-uniformclass and cost distributions: A case study in credit card fraud detection", *In Proc. of Knowledge Discovery and Data Mining*, 164–168,1998.

[13] R. B. Rao, S. Krishanan and R. S. Niculscu., "Data Mining for Improved Cardiac care", *ACM SIGKDD Exploration Newsletter*, vol.8, no.1, pp. 3–10, 2006.

[14] S. Tan, "Neighbor-weighted K-Nearest Neighbor for unbalanced text corpus", Expert *Systems with Applications,* vol. 28, no. 4, pp. 667–671, 2005.

[15] S. Tan, "An effective refinement strategy for K-Nearest Neighbor text classifier", Expert *Systems with Applications,* vol. 30, no. 2, 290–298, 2006.

[16] T. M. Cover P. E. Hart, "Nearest neighbor pattern classification", *IEEE Transactions on Information Theory*, vol. 13, pp. 21-27, 1967.

[17] T. Yang, L. Cao, C. Zhang, " A novel prototype reduction method for the *K*-nearest neighbor algorithm with $K \geq 1$" *In: Zaki, M.J., Yu, J.X., Ravindran, B., Pudi, V.(eds.), PAKDD 2010. LNCS*, Springer, Heidelberg, vol. 6119, pp. 89–100. 2010.

[18] W. Liu, S. Chawla, " Class confidence weighted knn algorithms for imbalanced datasets" *In: Huang, J.Z., Cao, L., Srivastava, J. (eds.) PAKDD 2011, Part II. LNCS*, Springer, Heidelberg, vol. 6635, pp. 345–356, 2011.

[19] X. C. Li, W. J. Mao, D. Zeng D, P. Su and F. Y. Wang, "Performance evaluation of machine learning methods in cultural modelling", *Journal of Computer Science and Technology*, vol. 24, no. 6, pp. 1010-1017, 2009.

[20] X. Wu et al., "Top 10 algorithms in data mining", *Knowledge Information Systems*, vol. 14, pp. 1-37, 2008.

[21] Y. Yang, T. Ault, T. Peirce, C. W. Lattimer, "Improving text categorization methods for event tracking", *In proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval,* pp. 65–72, 2000.

Best Practices in Measurements for Asset Characterization in Complex Engineering Systems

Giulio D'Emilia [1], **Diego Pascual Galar**[2], **Antonella Gaspari**[1]
[1]Department of Industrial and Information Engineering and of Economics, University of L'Aquila, L'Aquila, Italy
[2]Division of Operation and Maintenance Engineering, Lulea University of technology, Lulea, Sweden

Abstract - *The evolution of systems based on the integration of Internet of Things (IoT) and Cloud computing technologies requires resolute and trustable management approaches, in order to let the industrial assets, thrive and avoid losses in efficiency and, thus, profitability. In this work, a methodology based on the evaluation of the measurement uncertainty is proposed, able to suggest possible improvement paths and reliable decisions. Its application in field, for the identification of the vibration and acoustic emission signatures of highly-performing machining tools, allowed directing future actions for increasing the potentiality of a proper management of the information provided by measurements. In a complex scenario, characterized by a large number of devices and instruments, the compliance with the procedures for measurement accuracy proved to be a useful support. A careful strategy in the reduction of uncertainty is useful in particular for classification of real damages allowing mitigating the intrinsic high variability of natural defects and threshold of defects detectability can be maintained low.*

Keywords: Measurement uncertainty, condition monitoring, centerless grinding.

1 Introduction

Exciting opportunities are arising from the evolution of the systems in the current industrial scenario. The integration of Internet of Things (IoT) [1] and Cloud computing technologies [2], leads us the Internet of Everything (IoE), a network of networks where billions of objects are connected, providing an informative base without precedent [3]. This trend is based on a large amount of data and returns the same uncountable quantity of data, which is called "big data" [4] [5]. Suitable information gathered by this huge mass of data is traditionally extracted by means of the use of data mining techniques [6], artificial intelligent tools [7], decision support systems and knowledge-based systems.

The impact on the manufacturing companies is remarkable.

In past years, the development of enterprise information systems (EIS) have allowed management to give rise to an informative base useful to decision-making [8]. The ability of an organization to take all input data and convert them into knowledge is referred as Business Intelligence (BI) [9]. On the other hand, the development of industrial automation, in rapid evolution, helped the industries to increase their productivity, setting up complex control and monitoring systems, aiming at a real-time management of the activities and achieving a real competitive advantage [10].

Nowadays, the possibility of applying new intelligent technologies, by integrating the informative content coming from disparate devices seems to make this goal possible, through the convergence of the physical world and the digital world into the so-called cyber-physical systems [11]. This new trend for industrial systems is known as "Industry 4.0" or "Industrie 4.0", in German [12] [13]. The adoption of the methods typically used in engineering tasks underline the possibility of merging them with "good" measurements.

In this context, the international reference standards offer a basic support to set corporate activities into a well-defined framework, in terms of the requirements for the implementation of integrated management systems. Quality [14], Environmental [15] Energy [16] and Asset [17] management systems are perhaps the most known international standards that companies adopt for benchmarking, marketing and self-organization purposes. Nevertheless, the quality of the achievable results and outcomes cannot be granted automatically and tacitly, since many aspects that could lead to variability may occur, due to the structure of these systems themselves. From the informative flow point of view, each node of the network requires specific procedures for assuring proper exchange of information.

Like and more than other branches of knowledge, measurement topics are stressed, in order to compete to the compliance of this requirement. This is primarily because, we have to measure to get experimental, objective and reliable information and because measurement results cannot be provided in a deterministic and univocal way.

Although a scheduled approach appears to be promising, such as that proposed by the international reference standards, the methods and tools provided seem to be not exhaustive and not-completely decisive, in the upcoming new era that we are called to manage, due to the following main reasons:

--High-connectivity and big data. Large scale and complex systems call for standardized and univocal methods for collecting, analyzing and interpret heterogeneous data provided by different devices and platforms, which inevitably cause chaos due to the increasing use of multi-purpose but poorly tested devices [1].

--Proactive methods and models. New perspectives in sensor and sensors technologies arise, e.g. those connected to

mechatronic smart components [18]. Furthermore, new supply chain management techniques (e.g. 6C framework to connect stakeholders [19]), new IT solutions and process mining techniques must be involved and harmonized all.
--Multi-disciplinary skills. In order to maximize the fluidity of these activities, transversal competences and skills are needed (e.g. those referred to mechanics and electronics, control systems and automation, informatics, software and telecommunication engineering [20], [21]), able to work together and coherently.
--Reliability of information. As a basic goal of the work, refined methods of diagnosis and control, able to support the accuracy of data and information appear essential for data management and analysis and for parallel processing and storage strategies, with reference to the large amounts of incoming data [9], [22].
Measurements play a key role as far as for the presence of sensors and devices, for data processing techniques, for the informative flow control and as a common factor of several fields of knowledge.
The variability of information turns into measurement uncertainty [23]. Treating with uncertainty is not just a formal requirement; previous work of the authors, concerning the production process of high complexity components for automotive industry [24], demonstrated that uncertainty can be used as an engineering and management tool. Further, merging methodologies are set, sharing actions of well-established improvement approaches like six sigma [24].
A methodology based on the evaluation of the measurement uncertainty is then discussed in this work, to explain this goal. Three main guidelines are followed: the calibration as a guarantee given in advance, the uncertainty assessment of high-synthetic indicators and the adoption of an iterative approach during the process itself [25].
With reference to a condition monitoring (CM) application for a highly-performing centerless grinding machining tool, the preliminary calibration of the vibration and acoustic emission measurement chains and the reduction of the variability by improving the signal-to-noise ratio are used for the characterization of the asset state. Since the asset condition analysis is widely carried out using measurement data, attention to metrology concepts and in particular to calibration procedure will allow using the information provided by these data according to its own physical meaning. [24], [25], [26].
A condition monitoring application represents an important test-case for the methodology. In fact, both the identification of the machine signature is widely used in many applications, concerning for example bearing [27], or oil debris [28], and the operating scenario of interest is realized, in terms of the issues and challenges analyzed [29].
A careful strategy in the reduction of uncertainty will be demonstrated to be useful in particular for classification of real damages allowing to mitigate the intrinsic high variability of natural defects. It is expected to maintain low the threshold of defects detectability. In this application the sensors calibration is the concept particularly stressed.

This paper is organized as follows: in Section 2 the main steps of the methodology are briefly introduced and explained. Section 3 shows the application of the methodology to the test case selected. Short conclusions and future works end the paper.

2 Methodology

The methodology adopted is based on the measurement uncertainty evaluation and on its further reduction, in order to support actions for the improvement of decision's reliability. In this sense, the measurement uncertainty is seen as an efficacious engineering tool [25] [30].
In general terms, uncertainty is a measure of the 'goodness' of a result [31] and the approach considers the product and process conformity check [32].
The theoretical and experimental methodology is defined in 3 main steps, aiming at:
1) identifying the main uncertainty causes, taking into account real issues (STEP 1);
2) in field evaluating the uncertainty of measurement, merging all the most remarkable contributions (STEP 2);
3) defining improvement actions, able to reduce uncertainty and to recursively increase the reliability of decisions and provided solutions (STEP 3).
By iterating the afore-mentioned steps in a recursive manner over time, the methodology proposed is expected to provide an indication about the assessment of the performances to which it is referred, in line with the continuous improvement principle of many industrial management systems. In the following the above-mentioned steps for the methodology are briefly introduced.

2.1 STEP 1: Uncertainty causes identification

The first step of the methodology aims at the identification of issues connected to the validation of big amount of data. In facts, in order to start to use the measurement uncertainty, it is unavoidable to understand its causes. To this aim, some canonical techniques may be found in literature, such as those referred to the Procedure for Uncertainty Management (PUMA) [30]. In order to transfer the canonical techniques in field 4 main sub-steps have been identified, which are summarized as follows:

1.1 Characterization of the scenario,
1.2 Requirements and constraints,
1.3 Analysis's boundaries,
1.4 Materials,
  1.4.1 Literature survey,
  1.4.2 Test environment.

2.2 STEP 2: Uncertainty evaluation

The second step of the methodology aims at giving evidence to the measurement uncertainty in order to keep track of it, along all the informative flows in the process analyzed. Among the already existing methods that are usually followed for the measurement uncertainty assessment,

some alternative techniques may be also found in literature (e.g. partial derivatives, numerical calculation and Monte Carlo method), though they still remain quite complex and elaborate, from the processing point of view. Furthermore, the simplification does increase the uncertainty, unless the physical meaning of the operations is adequately taken into account [24]. To this aim, two sub-steps are identified, with reference to the specific field of application and taking into account the priorities emerging in the previous step (STEP1):

2.1 Uncertainty budget and test plan,

2.2 Indicators for measurement uncertainty.

2.3 STEP 3: Uncertainty reduction

Goal of this sub-phase is keeping a low level of uncertainty in order to increase the reliability of future action plans. Even at this stage, the state of the art suggests some approaches to keep control of the measurement uncertainty, such as the metrological confirmation in [33], or the conformity assessment in [34]. Taking advantage from these approaches, the following sub-steps are highlighted:

3.1 Measurement uncertainty results,

3.2 Remarks,

thus allowing for providing further indications not available preliminarily and contributing to the system's knowledge.

3 Test case

In this application, the effects of quality of measurements in condition monitoring (CM) analysis are evaluated. The analysis focuses on the behavior of components of a high performance center-less grinding machine, for the detection and localization of the defects on the ball screw, using a self-implemented classification algorithm and other pattern recognition algorithms [35]. Vibration and acoustic emission measurements are the input data for the analysis. The preliminary calibration of the measurement chains and the reduction of the variability by improving the signal-to-noise ratio are the methodological approaches mainly stressed. In the following, the methodology introduced in Section 2 is applied and each sub-step described in further detail.

3.1 STEP 1: Uncertainty causes identification

3.1.1 Characterization of the scenario

The use of condition monitoring approaches is increased over the last decades. Description, issues and possible solutions to deal with sensor-based condition monitoring and diagnostics applications can be widely found in literature, such as [36], [37] [38]. On the other hand, metrology techniques and information coming from measurement data offer a useful contribution in facing engineering issues (e.g. 5 M – Machines, Materials, Methods, Measurements, Modelling approach and 5 S - Sensing, Storage, Synchronization, Synthesis and Service [39]), all suggesting us to include sensors and measurement aspects properly, in a big data environment.

3.1.2 Requirements and constraints

Condition monitoring applications represent a suitable example of application, where experts are called to tackle with a large amount of data. In this context, the requirements and constraints mainly refer to the need of having available optimized solutions for:

- the exploitation of data coming from disparate sensors;
- the application of suitable data mining and data processing techniques;
- the possibility of assuring the reliability of results, with reference to both measurements and algorithms.

In particular, as far as for the measurements' reliability, this is intended as the possibility of adopting preventive solutions (in terms of rigorous methods and operating procedures, able to avoid or minimize errors from the beginning

3.1.3 Test environment

With reference to Fig. 1, the focus of the analysis is set on: (a) the Machine, (b) the Method and (c) the Measurement. In particular, the system of interest is preliminarily analyzed, in its mechanical and dynamic functionalities on the one hand, and in its software system and control unit and architecture, on the other hand, in order to identify both the machine typical failure modes and the quantities of interest to be monitored, to the ultimate aim of associating the physical meaning of the obtained indications (i.e. measurements), to the phenomenon in exam (i.e. failure modes, faults and damages).

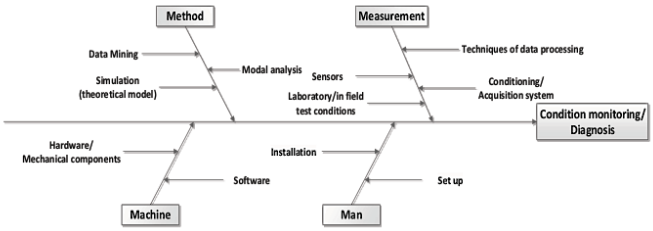


Fig. 1. Ishikawa diagram for a CM application

3.1.4 Test environment

The test environment refers to an axis test rig of a center-less grinding machine tool. It consists of a low-vibration granite machine bed and of a slide system [40]. A high-precision ball-screw system allows converting the rotary movement of the spindle drive into a linear movement of the unloading and infeed/recessing axes, which are rigidly connected to conveniently sized carriages sliding on the linear guidelines [41].

In this case, one of the most critical components, which cause machine breakdown, is the ball screw drive, since it determines the dimensional and surface precision of the work-pieces, depending on wear [35], [41].

The wear is due to the exchanged forces and solicitations, whose behavior is difficult to predict theoretically [42] [43] and for this reason the accuracy requirements satisfaction can be verified by means of an experimental monitoring [44].

Vibration and acoustic emission signals are used, in order to train the algorithms for machine learning developed in Matlab [26], [45].

3.2 STEP 2: Uncertainty evaluation

3.2.1 Uncertainty budget and test plan

Both real and artificial damages are taken into account, in order to apply and compare different machine learning techniques.

In order to guarantee the data accuracy, the following main aspects are taken into account:

- the most suitable positioning of sensors, in order to improve the signal-to- noise ratio of measurement data;
- the calibration of the vibration and acoustic emission (AE) measurement chains (absence of systematic error, repeatability and low level of noise). The analysis is carried out in laboratory and static conditions.
- the check of the quality of measurements when no remarkable damage is present on the system (reference conditions).

In facts, the willing of classifying also real damages requires a careful strategy in the reduction of the uncertainty, due to the intrinsic high variability of the characteristics of natural defects.

3.2.2 Indicators for measurement uncertainty

As far as for the **positioning of sensors**, several methods are studied, compared and selected in literature [46], [47] in order to assess the most suitable positions of the sensors on the machine. Here, sensor placement and frequency ranges to be examined for the condition monitoring application have been analyzed performing an experimental modal analysis, to identify the vibrations due to the resonances of the system and how the excited modal shapes look like [48], [49]. The results obtained in repeatability conditions are compared and qualitatively analyzed in terms of frequency and amplitude the Frequency Response Function (FRF) of each mode.

As far as for the **calibration** of the vibration measurement chain, the reference indicator to carry out the preliminary check of the vibration measurement chain is the sensitivity S expressed in V/g (where g is the acceleration due to gravity, g= 9,807 m/s2). The contribution of the measurement chain to the variability of the measurement data is assessed by means of mean value , and standard deviation s (Eq. 1, Eq. 2), calculated for each i-th estimated value of the sensitivity with i=1,…N (N=10 repeated tests) [51]

$$\bar{S} = \frac{\sum_{i=1}^{N} S_i}{N} \tag{1}$$

$$s = \sqrt{\frac{\sum_{i=1}^{N} (S_i - \bar{S}_i)^2}{N-1}} \tag{2}$$

As far as for the acoustic emission measurement chain, two acoustic emission sensors are mounted on the machine [35]. The positioning of the AE sensors may be optimized by means of preliminary analysis using FE models [51].

Repeated tests are carried out at a sampling frequency of 500 kHz. The RMS channels have been selected and used to acquire the signals and to check the absence of noise or of systematic error.

Finally, as far as for the **check of the quality of measurements in the reference conditions**, the variability bands ($low = \overline{RMS} - s_{RMS}$ and $up = \overline{RMS} + s_{RMS}$) for the sensors have been evaluated, when the machine realizes 5 different speeds (from 1000 mm/min up to 5000 mm/min), in steps of 1000 mm/min. The mean value of the Root Mean Square ($\overline{RMS}$) and standard deviation (s_{RMS}) of RMS values calculated on all the repeated tests, are evaluated according to the following equations (Eq. 3 and Eq. 4):

$$\overline{RMS} = \sqrt{\frac{1}{k}\sum_{j=1}^{k} v_{t_{RMS_j}}} \tag{3}$$

$$s_{RMS} = \sqrt{\frac{\sum_{j=1}^{k} (v_{t_{RMS_j}} - \overline{RMS})}{(k-1)}} \tag{4}$$

where $v_{t_{RMS}}$ stands for the RMS value of the measured signal v_t (Eq. 5):

$$v_{t_{RMS}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} {v_{t_i}}^2} \tag{5}$$

3.3 STEP 3: Uncertainty reduction

3.3.1 Measurement uncertainty results

With reference to the positioning of the sensors, the qualitative and pragmatic modal analysis allowed selecting the front wall of the guide system as one of the suitable position for the accelerometer. The positioning of the AE sensors is selected, being on the support of the ball nut of the ball screw system, where a high signal-to-noise ratio is assumable.

The sensitivity S obtained for the three measuring axes (x, y and z) of the accelerometer in the laboratory conditions described above, confirmed the calibration certificate indications. Since negligible changes in the sensitivity can be detected, the accelerometer measurement chain is positively verified, in terms of absence of systematic error and low level of noise, and it can be suitably used in field. To assess the functionality of the acoustic emission measurement chain, two AE sensors are mounted on the external sides of the ball screw nut by means of a magnetic support. The AE resulting level in static conditions of the machine results in the order of 0.0020 ± 0.0005 V, (with reference to a voltage full scale of 4V), being the noise really negligible.

The accelerometer is placed on the front wall of the guide, with x positive to the right, y positive upwards, and z parallel to the plane of movement of the slide. The variability bands of the indicator associated to each monitored speed for each

measuring axis (x, y, z) in positive (+) and negative (-) directions are evaluated. Figure 3 and 4 show the variability bands for the $\overline{RMS}$ attesting that that the vibrational signature is defined with very a low dispersion.

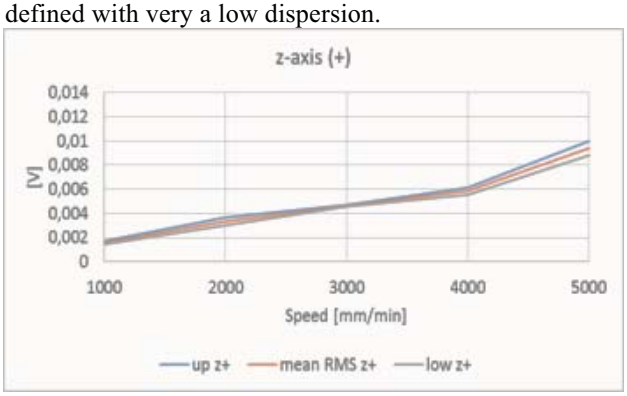


Fig. 3. Variability bands for mean RMS: z-axis measuring in positive (+) direction

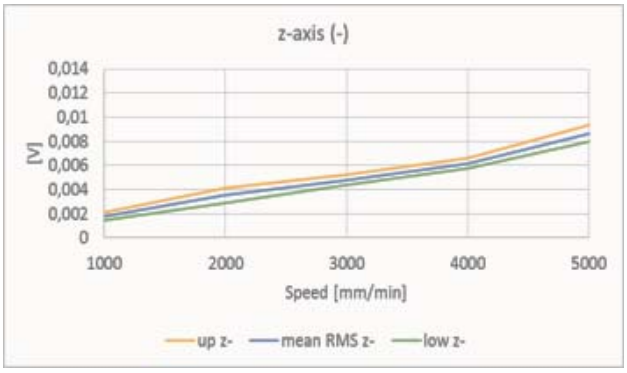


Fig. 4. Variability bands for mean RMS: z-axis measuring in negative (-) direction

Similarly, the trend of the same indicator associated to the different examined speeds together with their estimated variability bands are showed in Fig. 5 and 6, confirming that the AE measurement chains work properly also in working conditions and they can support the numerical procedures for defects classification.

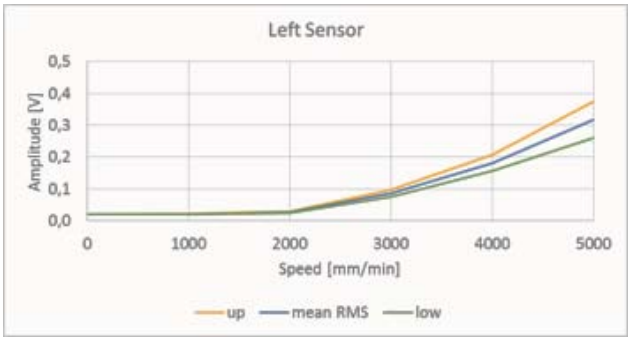


Fig. 5. Variability bands for mean RMS: left AE sensor

The obtained results show that the sensitivity threshold of the measurements to be intended as the maximum value of the indicator variability in correspondence of 0-2000 mm/min velocity range, is very satisfactory.

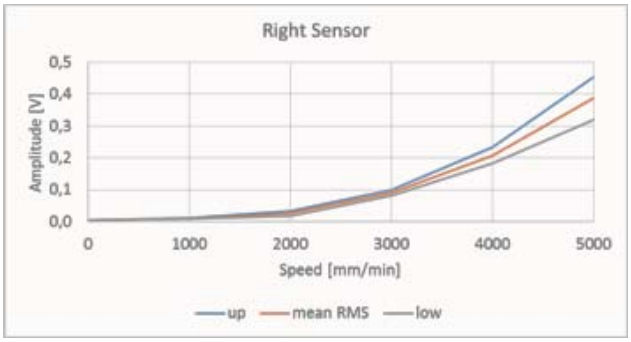


Fig. 6. Variability bands for mean RMS: right AE sensor

3.3.2 Remarks

The accuracy of measurement data is a pre-requisite to be assured as a priori condition, in order to support the success and the reliability of the condition monitoring analysis. In facts, the signature of the system, working in no damage conditions (i.e. the reference conditions), has been defined with a tiny uncertainty range, allowing identifying the occurrence of tiny variations of the working conditions with respect to the arise of defects

4 Conclusions

In this work, as a pre-requisite to be assured, particular attention has been paid to the preliminary calibration of the measurement chains and to the issues connected to the sensor positioning, allowing to define, with a very low level of variability, the working reference conditions for the system in exam.

The methodological approach adopted here is based on the validation analysis performed in advance, with respect to the further analysis for condition monitoring and diagnosis purposes (in terms of action taken for fault recognition, fault localization and identification of causes [52]) based on the evaluation of the measurement uncertainty (in terms of repeatability).

This solution appears suitable, in order to reduce the propagation of errors so that the threshold of defect detectability can be maintained low. This appears useful if defects have to be detected also at an early stage. Further, in order to detect incipient natural defects in complex systems, having a very low detectability threshold appears to be a mandatory requirement.

5 Acknowledgment

The authors would like to thank the Fraunhofer Institute for Production Systems and Design Technologies (Berlin, Germany), where part of the research work took place.

6 References

[1] I. Lee, K. Lee, "The Internet of Things (IoT): Applications, investments, and challenges for enterprises," Business Horizons, vol. 58, pp. 431-440, 2015

[2] M. Bayramustaa, V. A. Nasir, A fad or future of IT?: A comprehensive literature review on the cloud computing research, International Journal of Information Management, vol. 36, pp. 635–644, 2016

[3] A. Botta, W. de Donato, V. Persico e A. Pescapé, "Integration of Cloud computing and Internet of Things: A survey," Future Generation Computer Systems, vol. 56, pp. 684-700, 2016.

[4] S. F. Wamba, S. Akter, D. Gnanzou, A. Edwards e G. Chopin, "How 'big data' can make big impact: Findings from a systematic review and a longitudinal case study," Int. J. Production Economics, vol. 165, pp. 234-246, 2015.

[5] X. Jin, B. W. Wah e X. Cheng, "Significance and Challenges of Big Data Research," Big Data Research, vol. 2, p. 59–64, 2015.

[6] C. C. Aggarwal, "Data Mining - The Textbook", Springer, 2015.

[7] P. Galar, "Artificial Intelligence Tools", CRC Press Taylor and Francis Group, 2015.

[8] D'Emilia, G., Paolone G., Natale E., Gaspari A., Del Villano D., A Measurement-oriented Modelling Approach – basic concepts to be shared. Springer International Publishing, Software Technologies, Communications in Computer and Information Science 555, pp. 170-182

[9] M.-A. Aufaure, R. Chiky, O. Cure, H. Khrouf e G. Kepeklian, "From Business Intelligence to semantic data stream management," Future Generation Computer Systems, 2015

[10] B.R. Mehta, Y. Jaganmohan Reddy, Industrial Process Automation Systems – Design and implementation, Butterworth-Heinemann Ed., 2015

[11] K. Thramboulidis. A cyber–physical system-based approach for industrial automation systems, Computers in Industry, vol. 72, pp. 92–102, 2015

[12] C. Toroa, I. Barandiarana e J. Posada, "A perspective on Knowledge Based and Intelligent systems implementation in Industrie 4.0," Procedia Computer Science, vol. 60, pp. 362-370, 2015.

[13] S. Weyer, M. Schmitt, M. Ohmer e D. Gorecky, "Towards Industry 4.0 - Standardization as the crucial challenge for highly modular, multi-vendor production systems," IFAC-PapersOnLine, vol. 48, n. 3, p. 579–584, 2015

[14] ISO 9001:2009, "Quality Management Systems – Requirements"

[15] ISO 14001:2015, "Environmental Management Systems – Requirements"

[16] ISO 50001:2011, "Energy Management Systems – Requirements with Guidance for Use"

[17] ISO 55001:2014, "Asset management – Management systems – Requirements"

[18] D. Bradley, D. Russell, I. Ferguson, J. Isaacs, A. MacLeod and R. White, "The Internet of Things – The future or the end of mechatronics," Mechatronics, vol. 27, pp. 57-74, 2015

[19] K. Rong, G. Hu , Y. Lin , Y. Shi and L. Guo, "Understanding business ecosystem using a 6C framework in Internet-of-Things-based sectors," Int. J. Production Economics, vol. 159, pp. 41–55, 2015

[20] G. Hackenberg, C. Richterb and M. . F. Zäh, "A multi-disciplinary modeling technique for requirements management in mechatronic systems engineering," Procedia Technology, vol. 15, pp. 5-16, 2014

[21] M. M. Eskandar and V. Garousi, "Enegineering Control Software Systems: A Multi- Disciplinary Challenge," IEEE 978-1-4673-0750-5, pp. 1-6, 2012

[22] O. Kwona , N. Leea and B. Shin, "Data quality management, data usage experience and acquisition intention of big data analytics," International Journal of Information Management, vol. 34, pp. 387-394, 2014

[23] JCGM - 200:2012, "International vocabulary of Metrology-Basic and general concepts and associated terms," Paris

[24] G. D'Emilia, G. Di Rosso, A. Gaspari, A. Massimo, "Metrological interpretation of a six sigma action for improving on line optical measurement of turbocharger dimensions in the automotive industry", Proceedings of the Institution of Mechanical Engineering: Part D, Journal of Automobile Engineering, vol. 229 n. 2, pp. 261-269, 2015

[25] A. Gaspari, "Measurement-uncertainty: an innovative management tool in modern engineering systems - Validation of measurements within complex systems, supporting innovative technical and management strategies in current industrial revolution" Ph. D. Thesis, 2016

[26] D. P. Galar, Artificial Intelligence Tools - Decision support systems in condition monitoring and diagnosis, CRC Press - Taylor & Francis Group, 2015, p. 523

[27] Md. Mamunur Rashid, M. Amar, I. Gondal, J. Kamruzzaman, "A data mining approach for machine fault diagnosis based on associated frequency patterns" Appl. Intell., pp. 1-14, 2016

[28] Chuan Li, Ming Liang, "Enhancement of oil debris sensor capability by reliable debris signature extraction via wavelet domain target and interference signal tracking", Measurement, vol. 46, pp. 1442–1453, 2013

[29] D. Galar, A. Thaduri, M. Catelani, L. Ciani, "Context awareness for maintenance decision making: A diagnosis and prognosis approach", Measurement, vol. 67, pp. 137–150, 2015

[30] G. D'Emilia, A. Gaspari, "Measurement-uncertainty as an innovative managements tool in modern engineering systems" accepted for 14th IMEKO TC10 Workshop Technical Diagnostics, New perspectives in measurements, tools and techniques for system's reliability, maintainability and safety, Milan, Italy, June 27-28, 2016

[31] JCGM - 100:2008, "Evaluation of measurement data — Guide to the expression of uncertainty in measurement".

[32] ISO 14253-1:2013,, "Geometrical product specifications (GPS) – Inspection by measurement of workpieces and measuring equipment – Part 1: decision rules for proving conformity or nonconformity with specification"

[33] ISO 10012:2003, "Measurement management systems — Requirements for measurement processes and measuring equipment"

[34] ISO 14253-2:2013, "Geometrical product specifications (GPS) — Inspection by measurement of workpieces and measuring equipment — Part 2: Guidance for the estimation of uncertainty in GPS measurement, in calibration of measuring equipment and in product verification"

[35] G. D'Emilia, A. Gaspari, E. Hohwieler, A. Laghmouchi and E. Uhlmann, "Improvement of defect detectability in a amchine tools sensor-based condition monitoring application," proposed for publication on: International Journal of Advanced Manufacturing Technology, 2016

[36] R. Isermann, Fault-Diagnosis Systems - An Introduction from Fault Detection to Fault Tolerance, Berlin, Heidelberg: Springer-Verlag, 2006.

[37] A. Jablonski, T. Barszcz, "Validation of vibration measurements for heavy duty machinery diagnostics", Mechanical Systems and Signal Processing, vol. 38, pp. 248-263, 2016

[38] J. Lee, F. Wu, W. Zhao, M. Ghaffari, L. Liao, D. Siegel, "Prognostics and health management design for rotary machinery systems – Reviews, methodology and applications", Mechanical Systems and Signal Processing, vol. 42, pp. 314-334, 2014

[39] J. Lee, E. Lapira, B. Bagheri and H. Kao, "Recent advances and trends in predictive manufacturing systems in big data environment," Manufacturing Letters, vol. 1, pp. 38-41, 2013.

[40] I. D. Marinescu, M. Hitchiner, E. Uhlmann, B. W. Rowe and I. Inasaki, Handbook of machining with grinding wheels, Taylor & Francis Group, 2006

[41] A. Laghmouchi, E. Hohwieler, C. Geisert and E. Uhlmann, "Intelligent configuration of condition monitoring algorithms," Applied Mechanics and Material, vol. 794, pp. 355-362, 2015

[42] E. Uhlmann, C. Geisert, E. Hohwieler and C. Geisert, "Monitoring of slowly progressing deterioration of computer numerical control machine axes," Proc. of the Institution of Mechanical Engineers Part B Journal of Engineering Manufacture, vol. 222, no. 10, pp. 1213-1219, 2008.

[43] J. Z. Sobolewski, "Vibration of the ball screw drive," Engineering Failure Analysis, vol. 24, pp. 1-8, 2012

[44] H.-C. Mohring and O. Bertram, "Integrated autonomous monitoring of ball screw drives," CIRP Annals - Manufacturing Technology, vol. 61, pp. 355-358, 2012

[45] E. Uhlmann, A. Laghmouchi, E. Hohwieler and C. Geisert, "Condition Monitoring in the Cloud," Procedia CIRP , vol. 38, pp. 53-57, 2015

[46] R. Castro-Triguero, S. Murugan, R. Gallego and M. Friswell, "Robustness of optimal sensor placement under parametric uncertainty," Mechanical Systems and Signal Processing, vol. 41, p. 268–287, 2013

[47] P. Er, C. Teo and K. Tan, "Approach towards sensor placement, selection and fusion for real-time condition monitoring of precision machines," Mech. Syst Signal Process (article in press), 2015

[48] F. Bogard, K. Debray, Y.Q. Guo, "Determination of sensor positions for predictive maintenance of revolving machines", International Journal of Solids and Structures, Vol. 39, Issue 12, Pages 3159–3173, June 2002

[49] M. Brehma, V. Zabela, C. Bucherb, "Optimal reference sensor positions using output-only vibration test data", Mechanical Systems and Signal Processing, Volume 41, Issues 1–2, Pages 196–225, December 2013

[50] E. O. Doebelin, Measurement Systems, McGraw-Hill, 2004

[51] A. Cavuto, M. Martarelli, G. Pandarese, G.M. Revel, E.P. Tomasini, Train wheel diagnostics by laser ultrasonics, Measurement, vol. 80, pp. 99–107, 2016

[52] EN 13306:2010, "Maintenance – Maintenance terminology"

Strategies for distributed curation of social media data for safety and pharmacovigilance

Tim A. Casperson[1], Jeffery L. Painter[2], and Juergen Dietrich[3]
[1]tim.a.casperson@gsk.com, GlaxoSmithKline, RTP, NC, USA
[2]jeffery.l.painter@gsk.com, GlaxoSmithKline, RTP, NC, USA
[3]juergen.dietrich@bayer.com, Bayer, Berlin, Germany

Abstract— *We have developed a system for trained medical experts to* curate *patient authored text in social media posts (the process of manually reviewing posts to extract medical insights).*

The system provides annotation capabilities for medication names, events, indications, drug-drug interactions as well as documenting drug use and potential adverse events. This system is currently being used to create a gold standard training set for tuning and comparing the performance of adverse drug reaction detection classification methods.

The software tools facilitate the real-time, distributed annotation and curation of social media data for use by trained medical experts in support of annotating drug names and medical conditions in social media data.

Once complete, this will represent one of the largest collections of annotated social media text available for use in adverse drug reaction detection.

Keywords: social media; adverse event detection; classification

1. Introduction

Web-Recognizing Adverse Drug Reactions (WEB-RADR) is a groundbreaking European Union (EU) Innovative Medicines Innovation funded 3-year initiative to recommend policies, frameworks, tools and methodologies by leveraging these new developments to get new insights in drug safety [1]. Among WEB-RADR's many objectives is the ability to leverage social media for discovering new insights in drug safety.

While there are many on-going efforts to build automatic classification methods for ADR (adverse drug reaction) detection in social media [2][3][4], this project aims to garner one of the largest sets of hand curated data for the purpose of establishing a *gold standard* by which future, automated classification methods may eventually be compared. This is a critical component necessary, and still missing, from the public at large who are interested in cultivating the voice of the patient through social media to expand our capabilities in monitoring the safety of actively marketed drug products.

1.1 Project CRAWL

GlaxoSmithKline, Inc., a pharmaceutical company, initiated their own project investigating the use of social media for pharmacovigilance in early 2012, and that research developed into *Project CRAWL* (Contextualization of Real World Drug Use through Social Listening), working together with an external vendor to collect, aggregate and de-identify social media data. To support all of these efforts, a software tool called *Insight Explorer* was built to help *Project CRAWL* evaluate over 40,000 social media posts, covering a wide range of products and safety related questions [5].

Throughout the *Project CRAWL* initiative, our team worked with our GSK safety scientists to build a prototype application that would enable trained medical experts to manually evaluate social media data for various outcomes related to our own marketed products. The result of this prototype application was then presented to the IMI WEB-RADR team as a potential tool for their efforts.

An example of how the original version of the tool is used is shown in Figure 1 which included the ability for a GSK safety scientist to examine social media post content and meta-data to (1) determine whether it is applicable for a safety assessment or spam, (2) determine whether or not the poster talks about a particular product of interest (*in-scope* for review) or any other products, (3) capture basic demographic information about the poster or patient, and finally (4) to answer an array of questions about the data such as "is this patient seeking information?" or "is this a complaint about a product?".

When we learned of the WEB-RADR project, it was determined that the IMI would also require sophisticated tools to assist their team in meeting the primary goals of manually curating social media data in order to build a "gold standard" of curated *patient authored text*[1], and we launched a new project to make modifications to *Insight Explorer* to help with this effort. This paper reports on our findings to date in building a distributed tool for the curation of social media for a large scale project around safety and pharmacovigilance.

[1]We acknowledge that it is not necessarily the case that when a person posts on social media that the person is talking about their own experiences. Instead of belaboring the nuances of whether or not a social media poster refers to him or herself directly or to another (such as a friend or family member), we will simply refer to all of these social media posts as *patient authored text*.

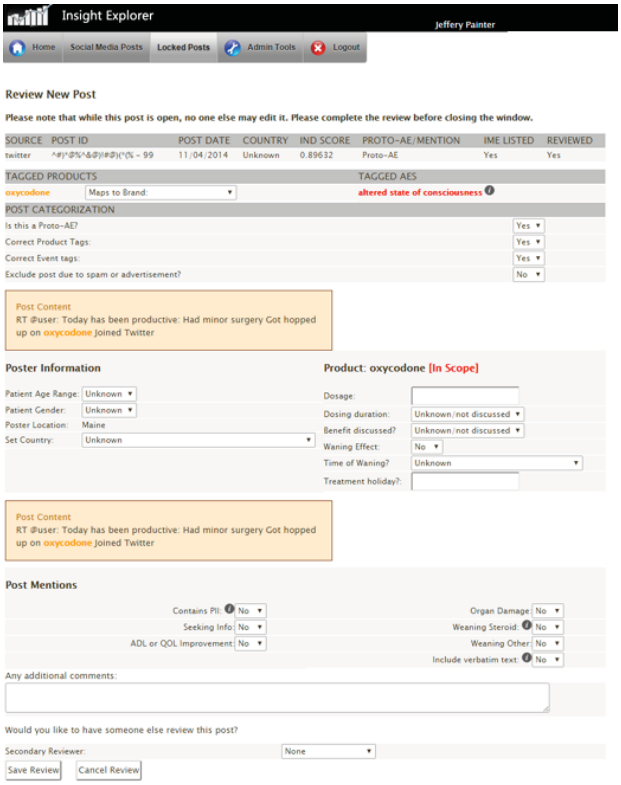


Fig. 1: Insight Explorer: Review Post

1.2 WEB-RADR Initiative

From the project charter, Johan Ellenius, UMC, explains it best:

"We are focusing on developing methods to automatically annotate medication names, Proto-AEs and unrelated events. One task in our work stream is to compare the classification performance of various methods for named entity recognition of these entities. In order to do the comparison, we must have a gold standard classification of the entities that we attempt to classify. So for example, in order to calculate the recall of our method for medication name annotation, we must be able to relate the correct identification of medication names with all occurrences of medication names in the tweets and posts. This is why we need a gold standard for medication names."

"The predictive models that perform the named entity recognition may draw on a lot of different information extracted from the posts. That is indeed part of the challenge, to identify purposeful features to use in the models. However, they don't need to be specifically gold standard classified, because the aim of our research is not to assess e.g. how well we are able to determine location of tweets or the sex of the author or something else that might serve as useful predictors in our models."

2. Background

Safety surveillance for adverse event (AE) detection encompasses a history which may well serve as a blueprint for the evolution of data mining in general. Since the discipline of pharmacovigilance first emerged in the 1960s, the strengths and weaknesses of this science have become quite evident. The early days of AE detection relied on the use of self-reporting from patients who experienced an unwanted side effect as well as their health care professionals through systems such as the FDA AERS[6] (adverse event reporting system) as well as homegrown solutions by each of the pharmaceutical companies [7].

The primary issue surrounding these self-reporting systems is that there was never evidence of the total population exposed to medications, and therefore, a much needed denominator to determine the likelihood of the event was missing [5].

Using observational data such as electronic health records and insurance claims data (where it is possible to compute a denominator) can supplement findings from spontaneous adverse event reports, but this type of data has additional problems with bias (e.g. indications are reported for insurance claims purposes rather than to indicate the exact patient experience).

In addition, access to these types of data typically require signing expensive license arrangements with individual data providers, and again, the population under study may be subject to underlying biases in the market these payers and providers serve (e.g. those who are insured are typically younger and healthier than the overall population, or a Medicaid or Medicare population which may be sicker than normal) [8].

By using social media sources, it is hoped that for the first time, it may be possible to capture the true voice of the patient in their own words. However, people do not typically speak the same language as health care professionals, and therefore, one must learn how to discern what patients mean when they express their experiences and interactions with health care systems through social media.

There is a need for a highly specialized social media curation tool for medical within the IMI (Innovative Medical Initiative). The IMI has kicked off a 3 year project which is partly for the study of publicly available patient authored text on social media. This project is called WEB-RADR. (WEB-Recognizing Adverse Drug Reactions)

The WEB-RADR project aims to supplement our current knowledge of patient authored text by manually curating thousands of social media posts gathered from Twitter and Facebook. The amount of data that the expert curators must process however required construction of new software tools and an infrastructure to help them distribute the workload evenly and minimize the risk of losing the valuable information they are collecting.

Early social media post curation was initially accomplished by using spreadsheets as the main tool. This process was tedious and not well suited for collaborative curation across a geographically distributed team.

Our team has custom designed our social media tool, *Insight Explorer*, for WEB-RADR's social media curation needs. This is a web based tool allowing remote access to a distributed curation team and is highly scalable.

3. Distributed Curation

The problem to be solved with distributed social media curation has many facets. Curation teams are scattered geographically into different time zones and need a customized system that enables them to collaborate and work remotely. In addition, if a team annotates social media posts in a traditional tool such as a spreadsheet, the data is very wide, is difficult to read, and is cumbersome to divide up posts between curators and collaborate across the team.

Social media posts need the information within it tagged and categorized into groups such as medication reported, brand name, generic name, medical events and indications reported such as flu or fever, as well as poster information if given, and any further comments from the curator. Furthermore, curators in our case are usually highly qualified medical experts. Any small improvements that can be made to their curation process that can save time will affect cost drastically.

Before the implementation of *Insight Explorer* as the WEB-RADR curation tool, curators would divide up posts equally among team members and place them in a spreadsheet for review. If the teams were to grow or shrink size once the curation process had begun, it would cause additional complications partly because it would be monotonous to reallocate the posts based on the new team size while ensuring that previously curated data was preserved.

Furthermore, the analysis of the data after curation would be more difficult because the data would need to go through the additional step of being read into an analytics platform. In *Insight Explorer*, the curation data is automatically saved in a relational database.

Our approach to this multi-faceted curation problem was to create a new version of *Insight Explorer*, specially customized to meet the needs of a diverse geographically distributed team of reviewers.

We worked with the WEB-RADR project leads to gather system requirements which have drastically increased the speed with which the teams can curate the data, as well as collecting all the results in a centralized data repository for analysis.

3.1 Incorporating Structured Terminologies

One of the requirements was to add a feature to enable the automatic look up and association between generic and brand name drugs. To accomplish this, we utilized the

Fig. 2: Brand Name and Generic Name are automatically looked up based on the reported term

Unified Medical Language System (UMLS) to develop our extracts of the *RxNorm* and MedDRA dictionaries using the UMLS MetaThesaurus [9] .

Much of our group's prior history was rooted in the development of tools and methods to exploit the use of standardized structured terminologies in support of safety activities [10], [11], [12]. As such, we have taken much of that history and knowledge and applied it to the development of tools such as this one to ease the curation process for our trained medical experts.

To assist our users from having to perform the unnecessary task of leaving the review screen in order to lookup codes from a standard terminology, we sought a method to incorporate them directly into the user experience to (1) improve the consistency of the annotations created by the expert reviews and (2) to reduce the time and effort of these valuable human resources from having to consult external tools to determine appropriate coding of the social media posts.

Therefore, we were able to add both a medical condition terminology (MedDRA) and a standardized drug catalog (RxNorm) to support these goals. The most recent version of the UMLS (2015AA) at the time the system was being developed was used to generate extracts of both the MedDRA terminology as well as RxNorm.

MedDRA was established in the late 1990's by the ICH (international Council for Harmonization of Technical Requirements for Registration of Pharmaceuticals for Human Use) to promote the sharing of medical information about medical conditions internationally. It serves as a standard terminology for classifying medical conditions into "Preferred Terms", also known as PT's (e.g. influenza) and Lowest Level Terms (LLTs such as flu) [17].

While MedDRA itself is a hierarchical terminology, we were only interested in the support of mapping user reported terms to lower level terms within the MedDRA ontology [13]. Since MedDRA then maps every low level term (LLT) to a preferred term (PT), those mappings are automatically displayed to the user once a match has been identified. If the expert reviewer knows the preferred term directly, they can also enter that and the corresponding LLT is automatically populated for them instead.

There were instances of confusion among curators which leave room for improvement in this area. In MedDRA version 17.1, not every LLT necessarily maps to a PT. One of our user's did experience this in the case of the term "locally advanced breast cancer", which is a valid LLT in

Fig. 3: Patient events are recorded and the patient reported text is mapped to a MedDRA term

MedDRA. The user expected that this term should map to the PT for “breast cancer", but it does not. Within the the UMLS concept hierarchy, the LLT was denoted within a single concept (CUI=C3495949), which did not have any corresponding PT codes. However, the UMLS does express a relationship between “locally advanced breast cancer" as being *classified_as* the PT for “breast cancer". At the moment, the current version of *Insight Explorer* is not taking advantage of these broader relationships, and is limited to more restrictive synonymy relation only.

Each of the relations between the LLT and PTs are stored within our MySQL database and loaded upon the review stage of the process to make lookup and retrieval appear in real time with minimal response gap to the end user.

This is similar to our process for mapping drug names automatically as well. In this case, we were able to extract from RxNorm each of the recognized brand names and map them directly to their generic counterparts. As an expert reviewer notes the patient authored representation of a drug concept, the reviewer copies that patient authored text directly into the interface, and then has the opportunity to attempt to describe the product either through the generic name or the brand name directly. If the generic is entered first, then a corresponding list of appropriate brand names is presented to the reviewer for selection as shown in Figure 2.

After identifying the potential medications that a poster may have mentioned, the curator is tasked with also trying to identify potential medical events related to the patient’s experience with the medication. These can be added dynamically through the “event editor" section of the review screen shown in Figure 3.

3.2 Poster and Patient Demographics

In addition to annotating the post itself, separate information about the poster and the patient (if the posters are referring to someone other than themselves) is annotated as well as show in Figure 4.

Here, the application allows reviewers to incorporate information given about the person including age, gender and location if that is determinable from the content of the post. While some location information may be provided as meta-data from some social media sources, this is not always reliable, and in some cases, may be at too detailed a level to comply with our de-identification process (e.g. some posts contain latitude and longitude coordinates which can pinpoint a post to a street address). For the work done prior

Fig. 4: Annotated poster and patient data

to the WEB-RADR initiative, these details were masked from our reviewers to prevent identifying a particular poster. We have employed methods to generalize coordinates to a zip code or state level where possible.

Gender identity may be inferred from the pronoun usage of a poster, or if they are speaking about a particular patient, from the language used to describe the person of interest (e.g. “my son" or “my daughter"). Additionally, a database of names has been utilized to help identify gender based on the username of the poster [14]. Again, the unique usernames have been de-identified prior to our loading of the posts into *Insight Explorer*, but the attributes are preserved for the reviewer indicating if the poster was male, female or unknown. However, our early studies have shown demographic data is generally difficult to infer. Gender is usually the easiest to identify, but age and other information is not as prevalent from our prior findings [15].

When a curator captures the medication and condition event information, the automatic term lookup pulls concepts for indications from MedDRA[16] version 17.1 (Medical Dictionary for Regulatory Activities). Our local version of MedDRA was extracted using the UMLS[9] MetaThesaurus.

When curators evaluate a social media post, they first review the verbatim text as entered by the poster, and then attempt to map that into a MedDRA term at the LLT and PT levels. *Insight Explorer* helps to ease this task by allowing the users to enter either a LLT or PT, and will then attempt to automatically map to the appropriate MedDRA entry, displaying the PT or LLT mappings automatically on the screen. This helps to insure that the curators are systematically entering data which conforms with the standardized terminology, and can later help in our knowledge discovery process to automate the mappings between patient authored text and a controlled vocabulary.

4. Deployed System

Insight Explorer in support of the WEB-RADR project for social media curation is deployed through Amazon Web Services running on Ubuntu Linux. From there, *Insight*

Explorer is accessed using a web browser after the user authenticates to the system. The application is compatible with most modern web browsers including Internet Explorer, Mozilla Firefox, and Google Chrome.

Adding additional curators are managed through the application by a system administrator. This is accomplished first by adding the users IP address to the Amazon hosted system's white list and then granting corresponding login credentials.

Administrators to the system have an additional security step of connecting through an encrypted SSH key exchange with the server. Passwords can only be incorrectly entered up to three times before accounts are locked which then requires that the account be unlocked by an administrator to enhance our security protocols and prevent tampering with the system.

4.1 System Architecture

The architecture of *Insight Explorer* follows a standard model-view-controller (MVC) framework. The presentation layer uses Apache Velocity templating language to render the page views in HTML and Javascript. The server tier is a Java™web application utilizing a MySQL database instance for persisting the social media reviews by each curator.

4.2 Team Collaboration

The four expert curators were initially divided into two teams of two. When both teams agree on a social media post, the post will be considered as "gold" and marked final in the database for classification as such.

When two teams disagree on their annotation of a particular post, then a super user team determines the final resolution of the disagreement. The decision of the super user team will determine the outcome to be marked finally as "gold." The super user team can be much smaller because the number of posts to be reviewed is estimated to be only 15% of the total. Based on prior research, we expect the teams to agree 85% of the time [18].

A flow diagram demonstrates how the review process occurs and the steps taken to reconcile differences between the review teams by a super user shown in Figure 5. The diagram outlines that we have multiple instances of *Insight Explorer* running, each possessing its own unique database instance for recording the expert's annotations of the social media posts. Each team reviews all of the posts selected for this ADR detection experiment.

To preserve the process, each team will review all of the posts in exactly the same order. Once the user begins the review process, a social media post is selected from the database and presented to the reviewer, at which point he or she will go through the following steps: (1) read the post, (2) identify relevant text, and (3) map those terms for drugs and conditions.

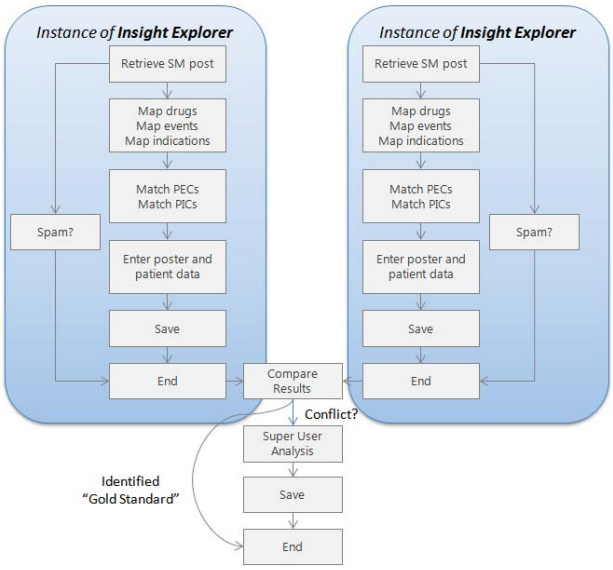


Fig. 5: Process flow supporting parallel teams using multiple instances of our tool

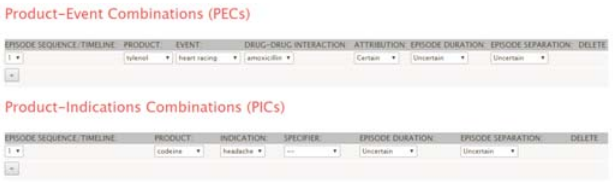


Fig. 6: Event modeling for product indications and events in social media

4.3 Modeling Complex Relations

Once a reviewer has identified the drugs and conditions in a post, then the reviewer next evaluates the post to determine if there are any complex events, such as product-event combinations (PECs) or product-indication combinations (PICs) present. The reviewer can then build those relations based on the drugs and conditions they identified above as shown in Figure 6.

A product event combination occurs when it is discernible from the post that a patient experienced an adverse event directly related to taking a product. A product-indication combination (PIC) is when a post definitively provides context that the patient and/or poster is taking the product to treat a specific medical condition (i.e. the indication) for that drug.

When curators tagged an associated PEC for a post, social media posters claimed about 35% of the time that a medication listed in their post caused an adverse event (such as loss of appetite or insomnia). The other 65% of the time it was uncertain whether the poster claimed a medication referenced in the post caused the adverse event or not. Posters almost never mentioned how long the duration that

the adverse event occurred.

When curators tagged an associated PIC for a post, the posters mentioned about 25% of the time the indication (e.g “psoriasis" or “epilepsy") that they were taking the medication for.

Again, the PEC and PIC annotations will ultimately go through the review process and the outcomes of these complex relations will be resolved by the super user if the curation teams are in disagreement on the models.

5. Discovery

Post curation has been underway for several months, and at the time of this writing, the system contains approximately 60,000 social media posts with the potential for more to be added. The team is currently reviewing social media data for six products of interest to build the gold standard. At this point of the curation process, more than 90% of the content has been marked as spam. Initially, a small team was curating posts only part time, and they averaged just a few thousand posts per month.

In the interest of time, each team was increased to approximately 10 members, with some of them being recruited full time to help complete the project sooner. The system is flexible enough to allow teams to grow and shrink as needed with minimal impact on the curation process while maintaining consistency in the reporting results.

Less than 500 of the 10,000 posts reviewed were not marked as spam. Of those 500 posts, 23 unique events have been recorded (associated with a MedDRA PT) and 132 unique indications have been mapped to MedDRA PT codes from the patient authored text.

The Web-RADR team did not initially want to filter spam programmatically since they wanted to be certain they were reviewing all posts that might have mentioned one of the products of interest in the study. However, due to time constraints, and the large volume of spam found in the data, we have adapted the system to automatically tag spam to speed the curation process. A spam filter was designed in an earlier version of *Insight Explorer* for internal use by the GSK team which was subsequently applied to the WEB-RADR project.

The majority of posters did not specify if they were the patient. However, when they did specify, the poster usually was the patient. To date, the reviewers have found only a few posts that were designated as either friend of the patient, patient health care provider, family, or simply not the patient. Otherwise, the reviewers mark them as an “unknown" poster type.

6. Future Direction

Insight Explorer has proven itself as a solid platform for the annotation and curation of social media data for safety and pharmacovigilance. We hope to expand on the software’s capability to incorporate some of our group’s other efforts around automated trend and topic analysis. Combining these facets of text mining to the annotation process would lead us toward developing a system by which we can eventually understand the strengths and weaknesses inherent to social media data as applied to ADR detection and other safety related activities – to understand truly the voice of the patient in a way that was previously inaccessible to the pharmaceutical industry.

To further this aim, we are exploring the tool’s capability to deal with longitudinal data through extractions of threaded discussions in online patient forums, as well as making the tool more flexible for trained medical experts to define their own protocol of questions that they may wish to ask of the data without having to spend as much time customizing the tool for each particular task.

7. Conclusions

Although WEB-RADR is still early in the process of curating thousands of social media posts, once completed, this will represent one of the largest collections of annotated patient authored texts available for furthering our knowledge and understanding of how the patient’s experiences with medicines are communicated through social media networks.

This will add a valuable resource to measure the performance of machine learning algorithms in their ability to automatically tag and predict adverse drug reactions in social media.

Additionally, the research initiated by the GlaxoSmithKline Safety Listening Laboratory, aims to develop specific classification algorithms around topics of interest beyond ADR detection, and this data set will help to further those goals as well.

8. Acknowledgments

The authors wish to thank the IMI WEB-RADR project leads and Greg Powell (GSK) for arranging the connection between the developers of *Insight Exporer* and the WEB-RADR team. We would also like to acknowledge the work performed by Epidemico for data collection and anonymization which was gathered based on the key word selections of products identified by the WEB-RADR team. We extend thanks as well to Magnus Lerch (Bayer) and Antoni Wisniewski (AstraZeneca) for working to develop the user requirements and giving feedback on test versions of the software.

Also, thanks to Grant Thomson of GSK for providing graphics support. Final thanks to GlaxoSmithKline for generously agreeing to donate the Insight Explorer software to the WEB-RADR project. The GSK Safety Listening Lab is actively pursuing continued research in the field of social media use for pharmacovigilance efforts and is open to collaboration opportunities with other research institutions in the field.

References

[1] R. Ghosh and D. Lewis, "Aims and approaches of web-radr: a consortium ensuring reliable adr reporting via mobile devices and new insights from social media," *Expert opinion on drug safety*, vol. 14, no. 12, pp. 1845–1853, 2015.

[2] C. C. Yang, H. Yang, L. Jiang, and M. Zhang, "Social media mining for drug safety signal detection," in *Proceedings of the 2012 international workshop on smart health and wellbeing*. ACM, 2012, pp. 33–40.

[3] A. Patki, A. Sarker, P. Pimpalkhute, A. Nikfarjam, R. Ginn, K. O'Connor, K. Smith, and G. Gonzalez, "Mining adverse drug reaction signals from social media: going beyond extraction," *Proceedings of BioLinkSig*, vol. 2014, pp. 1–8, 2014.

[4] R. Feldman, O. Netzer, A. Peretz, and B. Rosenfeld, "Utilizing text mining on online medical forums to predict label change due to adverse drug reactions," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2015, pp. 1779–1788.

[5] Powell, Seifert, Reblin, Burstein, Blowers, Menius, and Painter, "Social media listening for routine post-marketing safety surveillance," *Drug Safety*, pp. 1–12, 2016.

[6] FDA, *FDA Adverse Event Reporting System*, 2016 (accessed Feb 8, 2016), http://www.fda.gov/Drugs/GuidanceComplianceRegulatoryInformation/Surveillance/AdverseDrugEffects/.

[7] S. J. Reisinger, P. B. Ryan, D. J. O'Hara, G. E. Powell, J. L. Painter, E. N. Pattishall, and J. A. Morris, "Development and evaluation of a common data model enabling active drug safety surveillance using disparate healthcare databases," *Journal of the American Medical Informatics Association*, vol. 17, no. 6, pp. 652–662, 2010. [Online]. Available: http://jamia.oxfordjournals.org/content/17/6/652

[8] W. Hersh, M. Weiner, and P. Embi, "Caveats for the use of operational electronic health record data in comparative effectiveness research," *Medical care*, vol. 51, no. 8, pp. S30–S37, 2015.

[9] Aronson and Lang, "An overview of metamap: historical perspective and recent advances," *Journal of the American Medical Informatics Association*, vol. 17, no. 3, pp. 229–236, 2010.

[10] J. L. Painter and N. L. Flowers, "Codeslinger: An interactive biomedical ontology browser," in *Artificial Intelligence in Medicine*. Springer, 2009, pp. 260–264.

[11] G. H. Merrill, P. B. Ryan, and J. L. Painter, "Construction and annotation of a umls/snomed-based drug ontology for observational pharmacovigilance," *Methods*, 2008.

[12] J. L. Painter, K. M. Kleiner, and G. H. Merrill, "Inter-translation of biomedical coding schemes using umls," in *AAAI Fall Symposium*, 2006.

[13] G. H. Merrill, "The meddra paradox," in *AMIA annual symposium proceedings*, vol. 2008. American Medical Informatics Association, 2008, p. 470.

[14] M. Kantrowitz, "Name corpus: List of male, female, and pet names," http://www.cs.cmu.edu/afs/cs/project/ai-repository/ai/areas/nlp/corpora/names/0.html, 1994 (accessed 25-March-2015).

[15] R. L. DiSantostefano, J. L. Painter, M. Thomas, and G. Powell, *Safety Assessment and Selection Bias: Who uses social media to communicate about medications?*, August 2015, iCPE, Boston, MA (Poster session).

[16] I. F. of Pharmaceutical Manufacturers, "Meddra (medical dictionary for regulatory activities)," http://www.meddra.org, 2015 (accessed June 1, 2015).

[17] Brown, E. G., L. Wood, and S. Wood, "The medical dictionary for regulatory activities (meddra)," *Drug Safety*, vol. 20, no. 2, pp. 109–117, 1999.

[18] D. L. MacLean and J. Heer, "Identifying medical terms in patient-authored text: a crowdsourcing-based approach," *Journal of the American Medical Informatics Association*, vol. 20, no. 6, pp. 1120–1127, 2013.

Gene Selection from Microarray Data for Age-related Macular Degeneration by Data Mining

Yuhan Hao and Gary M. Weiss

Abstract—**A DNA microarray can measure the expression of thousands of genes simultaneously. Previous studies have demonstrated that this technology can provide useful information in the study of molecular pathways underlying Age-related Macular Degeneration (AMD) process. Relevant genes involved in AMD are not understood completely. The microarray dataset used in this study contains 175 samples measured around 41,000 genes' expression. The samples have two classes, normal eyes and AMD eyes. Dimensionality reduction from 41,000 features is necessary before a classifier for distinguishing normal and AMD eyes can be built. In this paper three established methods are utilized to perform feature selection: Naive Bayes with feature removal, Logistic regression with L1-regularization, and Decision Tree methods (the resulting classification accuracies are 89.66%, 77.01% and 66.67%, respectively). The microarray dataset is also visualized using Principal Component Analysis (PCA). The PENK and TRAF6 genes are the only two genes which are selected by the three feature selection methods and we expect that both of them are involved in the extracellular signal transduction. Functional Annotation Clustering of genes selected by any classifier also suggests that most genes are responsible for signal transduction in individual cell and between cells. The signal transduction pathway containing the selected genes may play an important role in AMD pathogenesis .**

1. Introduction

Age-related macular degeneration is a progressive neurodegenerative disease, which primarily affects retina pigmented epithelium (RPE) cells in the retina. RPE and choroid tissues contribute to maintaining vision function. RPE layers will eliminate the shedding outer segment of photoreceptors and promote retinal adhesion stabilizing alignment. Dysfunction of RPE usually results in disruption of retinal adhesion in persistent retinal detachment or photoreceptor apoptosis [1]. Nearly 40% of people over 75 years old have some pathological signs of AMD [2]. The molecular pathogenesis of AMD is not fully understood. A microarray can measure the expression of thousands of genes simultaneously, so it allows us to analyze this disease by a top-down approach.

This study uses the microarray dataset on Gene Expression Omnibus, Series GSE29801 [3]. It contains 175 RPE tissues samples from normal people and AMD patients. Although this dataset contains information about gender and age, only gene expression data is used in this study, because the goal is to discover representative genes for AMD molecular pathogenesis. Because the microarray data consists of 175 samples with 41,000 genes as features, dimensionality reduction is required before the data can be mined to generate a classifier capable of distinguishing normal samples from AMD samples. One straightforward method for reducing the dimensionality is to set the normal samples as standard and calculate difference from AMD samples and the p-value from GEO2R by unpaired two-tailed t-test with unequal variance [4]. The p-value is adjusted by permuting class labels 1,000 times with the Fisher-Yates methods [5]. GEO2R is a web tool that is able to compare groups of samples in order to identify genes that are differentially expressed across two experimental conditions.

In this paper three established feature selection algorithms are used to select relevant genes. The first method, Naive Bayes with feature removal, tends to select the features that maximize the accuracy of the generated classification model. The second method, logistic regression with L1 regularization, forces most coefficients to zero and the features with large coefficients represent the selected features. In both cases the selected features represent a relevant-gene list. These two classification methods are generative methods that can learn non-linear boundaries. The third method that is used is a decision tree method. This method is very robust with respect to large numbers of features, and can build an effective classifier that only utilizes the most informative feature. That is, decision tree methods automatically identify the most relevant features and ignore irrelevant or redundant features. The features that appear in the decision tree form the relevant-gene list. Finally, in addition to these three methods, an ensemble feature selection method is used to aggregate the selected features from these three methods [6].

Only two genes, PENK and TRAF6, are selected by all three methods, while 36 genes are selected by one of the three methods. We believe that the selected genes, especially PENK and TRAF6, will make a contribution to molecular pathway in the process of AMD and give hints to future therapies for this disease.

The purpose of this study is not to propose a new feature selection algorithm that can help build an effective classifier, since we focus on the molecular mechanism of AMD instead of diagnosis of this disease. Rather, the goal is to utilize data mining approaches to investigate biological systems and try to identify genes hidden in the microarray data that may be important to the AMD process.

The rest of the paper is organized as follows. In Section 2 we describe the microarray dataset, the four feature selection methods, a method to visualize the dataset, and one bioinformatics tool. Section 3 describes the experiments and results. Related work is discussed in

Y. Hao is with Fordham University, Bronx, NY 10458 USA (e-mail: yhao2@fordham.edu).

G. M. Weiss is with Fordham University, Bronx, NY 10458 USA (e-mail: gaweiss@fordham.edu).

Section 4 and the conclusion is provided in Section 5.

2. Materials and Methods

This section describes the microarray dataset and the methods and tools used in this study. The dataset is described in Section 2.1, the four feature/gene selection methods are described in Sections 2.2 to 2.5, the use of PCA for visualizing the dataset using three values is described in Section 2.6, and a functional clustering method to cluster similar genes is described in Section 2.7.

2.1 Microarray Dataset

This microarray dataset of Newman et al. [3] is measured from RNA collected from eyes of the human. It consists of two types of samples, 96 from RPE/choroid of normal eyes and 79 from RPE/choroid of AMD eyes. The training dataset has 88 samples which are selected randomly (48 normal eyes and 40 AMD eyes). Each sample has 41,000 features or genes. The test dataset consists of 48 normal eyes and 39 AMD ones.

2.2 Naive Bayes Classifier and Feature Removal

The Naive Bayes classifier is used to classify eyes status as normal or AMD. It assumes that continuous features are independent given class and normality. The assumption of independence does not hold for this dataset since genes are highly correlated. The data is transformed by applying the log_2 function in order to have the data more closely resemble a normal distribution. The class is determined by joint likelihood, $P(Gene_1, Gene_2, Gene_3.../ Y)$, $Y \in \{normal, AMD\}$.

Feature removal is based on maximal test accuracy. It starts with $F=\{x_1, x_2, x_3,...\}-\{x_i^1\}$, which x_i is removed Gene i and x_i satisfies argmaxi accuracy(F). Then it enters a loop, finding higher test accuracy for removing another attribute, x_i^2: argmaxi accuracy(F), removing x_i^2 to F feature with highest accuracy increase: $F=F-\{x_i^2\}$. The loop is repeated until test accuracy ceases to increase. The attributes that exist in F are the selected genes.

2.3 Logistic Regression with L1 Regularization

The Logistic Regression classifier is appropriate when dependent variable is nominal and the independent variable is continuous. The sigmoid function is used to represent the likelihood of the class. The learning algorithm is as follow:

$$w_j \leftarrow w_j + \Delta w_j - \frac{sign(w_j)}{\lambda} \qquad (1)$$
$$\Delta w_j = \eta * x_j^i * [y^i - g(w^T x^i)]$$

The learning step, η is set to 0.0001 and the number of iterations is set to 10,000. Meanwhile, in order to find the most representative or important genes, Maximum A Posteriori L1-regularization is used to force some parameter values to zero, equivalent to the function of feature selection. In this case, λ is equal to 1/500, a bias to minimize the number of non-zero variables.

2.4 Decision Tree (J48)

A decision tree is an expressive algorithm that generates a tree-like graph with leaf nodes and edges and can handle large numbers of features. The decision tree algorithm will tend to ignore redundant and irrelevant features. The J48 decision tree algorithm, which is part of the WEKA data mining suite, is used in this study. The splitting attributes that appear in the induced decision tree classification model are the genes selected by the model.

2.5 Ensemble Feature Selection

To increase the confidence of feature selection results and decrease the bias and errors induced by unsatisfied assumptions of the model, an ensemble feature selection technique [7] is employed. Ensemble feature selection methods utilize the same basic idea of ensemble classification algorithms [8]. A number of different feature selection algorithms are used and then the selected features from each algorithm are then aggregated. This study simplifies the ensemble feature selection process, which typically ranks each selected feature and then accepts only those with a high overall ranking. However, the ensemble feature selection in this study only focuses on the intersection and union of features selected by Naive Bayes, Logistic Regression and Decision Tree methods.

2.6 Principal Component Analysis (PCA)

PCA is a widely used statistical method to decrease dimensionality. In this study it is not used as a classifier, but as a method to simplify the visualization of the data. PCA is applied to the entire dataset (training and test data) and then the first three principal component (PC) vectors are selected as the *x*, *y* and *z* values to be plotted.

2.7 The Database for Annotation, Visualization and Integrated Discovery (DAVID)

DAVID is interactive web software that is used to cluster genes by function annotations. It provides a rapid approach to reducing large lists of genes into functionally related categories, which are ordered by enrichment of the category. It is widely used among researchers from over 5,000 institutes globally and cited by more than 6,000 academic publications [9]. In this study, the input is the union of the genes selected by the three base feature selection methods. The output is Functional Annotation Clustering Report. It shows different functional categories where genes most enriched. (DAVID: david.ncifcrf.gov)

3. Results

In this section the microarray data is filtered and we obtain 86 features and transform the expression of the gene. We then select relevant genes using the three base feature selection methods and aggregate the relevant genes using ensemble feature selection. Finally, we visualize the dataset using PCA

3.1 Filtering of AMD microarray dataset

GEO2R provides a way to calculate the p-value of each gene which is used to remove genes with random changes of expression (Table 1). The genes in Table 1 are listed in the order according to their adjusted p-value. The genes with large p-value indicate that their expression does not have a stable change between two groups, and they will

lead to more variations in the model. The threshold of adj p-value 0.05 is chosen. The genes with higher 0.05 adj p-value are removed. Also, it is found that some genes are repeated several times and some lack of annotation information. Finally, 86 genes are left and take on the role of features for the classification task.

3.2 Naive Bayes classifier for Eye Status Classification

In Naive Bayes classifier, the log_2 transformation is needed for the microarray expression data, continuous variable. The distribution of the transformed dataset is closer to a normal distribution, so it is fair that we use normal distribution function to estimate $P(Gene_i/Y)$.

The accuracy of this model is 83.91%. Then feature removal method is used, which tries to achieve maximal test accuracy. There are 20 genes left and they are listed in Table 2. Genes 74 to 86 are almost selected continuously. The test set accuracy, somewhat surprisingly, increases to 89.66%.

3.3 Logistic Regression for Eye Status Classification

The training dataset for Logistic Regression has added one constant column where each cell is 1 working as a constant. In order to find the most important genes in this AMD dataset, MAP with L1 regularization (η=0.0001, λ=1/500 and the number of iterations is 10000) is used to get minimal number of non-zero variables. The accuracy of the model is 77.01% and the time to build this model is 2 minutes. A larger learning rate, η and smaller iterations are also tested, but decreases the accuracy to around 50%. The value of weight vector provides a way to determine which genes are relatively dominant in this classification. Table 2 also shows that 16 attributes are selected by logistic regression and weight of these 16 attributes are higher than 0.5 or lower than -0.5.

TABLE 1
GENES ORDERED BY ADJ. P VALUE FROM GEO2R

Gene ID	adj.P.Val	Gene.symbol
30870	0.00014	----[a]
10463	0.00197	DNM3OS
9571	0.00533	----
29804	0.00533	AKAP8L
.	.	.
44181	0.85117	----
2160	0.85117	PFN1
9502	0.85117	EPS15
15494	0.85117	YLPM1
.	.	.
9773	0.99985	LRRC72
37141	0.99985	----
38929	0.99985	DZANK1

This table only contains the small fragment of the whole dataset.
[a]Information of Gene symbol is not complete.

3.4 Decision Tree for Eye Status Classification

The J48 decision tree algorithm is used to classify normal and AMD eyes. The accuracy of the induced decision tree, shown in Fig. 1, is 66.67%. This accuracy is a little lower than for the other two classifiers, but is higher than the 52.87% accuracy of the ZeroR method, which corresponds to predicting the majority class. The splitting attributes in Fig. 1 comprise the selected genes (Table 2).

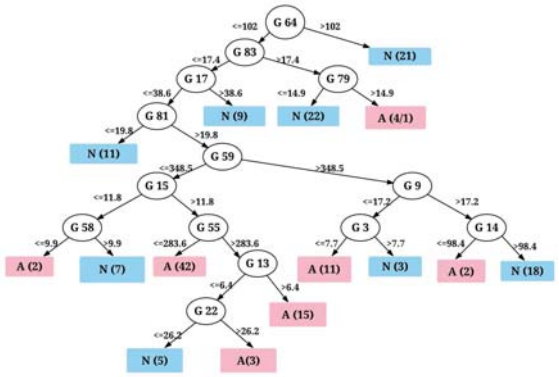


Fig. 1. J48 decision tree of classification of normal and AMD eyes. The rectangle leaf nodes represent the class of the sample and round splitting features represent selected genes. Accuracy of this tree is 66.67%. The following abbreviations are applied: G--Gene, A--AMD, N--Normal.

3.5 Ensemble Feature Selection for Three Classifiers and DAVID

Ensemble feature selection can reduce bias and variance and increase confidence. In this study, the ensemble feature selection contains three classifiers which are used to pick up candidate genes. Table 3 shows that information of genes which is selected at least twice and the first two genes, PENK and TRAF6, which are selected by all three classifiers. Moreover, there are 36 genes that at least are selected by one classifier.

Functional Annotation Clustering by DAVID is used for these 36 genes to discover what functional clusters these genes belong to. Add the gene list to DAVID and it generates numbers of functionally related categories ordered by enrichment. The first three functional categories are about extracellular region, negative regulation of cell

TABLE 2
RELEVANT-GENE LIST FROM THREE CLASSIFIERS

Naive Bayes	Logistic Regression	Decision Tree
3	7	3
29	12	9
34	14	13
46	15	14
58	27	15
62	29	17
66	31	22
71	34	55
73	35	58
74	44	59
75	58	64
77	59	79
78	62	81
79	79	83
80	85	----
82	86	----
83	----	----
84	----	----
85	----	----
86	----	----

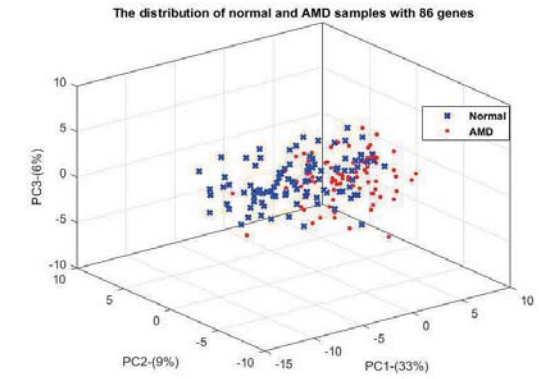
communication, and cell surface receptor linked signal transduction. All three clustering functions indicate those genes are involved in the process of signal transduction.

3.6 *Visualization of the dataset using PCA*

Principal component analysis is a common technique to reduce the dimensionality of dataset by mapping variables to a new set of variables. It has been used to analyze gene expression patterns [10]. Even though PCA is used to cluster genes in the whole dataset, instead of being a classifier, it also can be used for gene selection in cancer microarray dataset [11]. In this study it is used to illustrate the distribution of samples with 86 genes and 12 genes which are selected by two classifiers (Table 3).

In Fig. 2 the *x*, *y* and *z*-axis represent PC1, PC2 and PC3 respectively. Figure 2a indicates the first, second and third principle components contain 33%, 9% and 6% of the total information. However, in the case of selected genes (Fig 2b), PC1 can already represent 97% of total information.

(a)



(b)

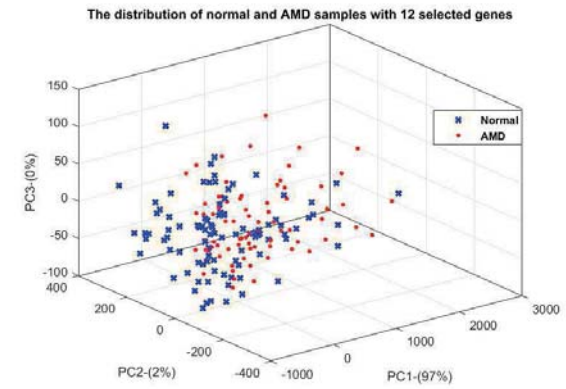


Fig. 2. Gene expression of normal and AMD eyes. It is established by PCA and x,y,z represent PC1, PC2 and PC3 respectively. Blue crosses represent normal eyes, and red dots represent AMD samples. The samples are not successfully separated under 86 genes, but PC1 under 12 selected genes can represent 97% of data. It indicates that selected genes are effective for classification of normal and AMD eyes.

These results show that the total dispersion in 86 genes PCA is much larger than in the 12 selected genes by any two classifiers. It strengthens the effectiveness of selected genes for classification.

TABLE 3
RELEVANT GENES SELECTED BY ENSEMBLE FEATURE SELECTION

Gene ID	Gene.symbol	Gene.title
58[a]	PENK	proenkephalin
79[a]	TRAF6	TNF receptor-associated factor 6
3	KATNAL2	katanin p60 subunit A-like 2
14	CLUAP1	clusterin associated protein 1
15	FAM208B	family with sequence similarity 208, member B
29	WASH1	WAS protein family homolog 1
34	TTC12	tetratricopeptide repeat domain 12
59	LINC00674	long intergenic non-protein coding
62	HIST1H3D	histone cluster 1, H3d
83	CYB5R1	cytochrome b5 reductase 1
85	CD163L1	CD163 molecule-like 1
86	MYRIP	myosin VIIA and Rab interacting protein

Those genes are selected at least by two classifiers.
[a]Only two genes are selected by all three classifiers.

4. RELATED WORK

Naive Bayes has been used in the analysis of microarray data. Though the independence assumption is an obstacle for Naive Bayes, it can be addressed by using Bayesian hierarchical models, which account for biological associations in a probabilistic framework. But for unknown interaction between genes, we still have to assume independence [12]. Logistic regression with lasso or L1-regularization is commonly used to handle high-dimensional data. Adaptive Lasso containing another term $1/w_j$ to control lasso strength [13], and elastic-net method [14] combing L1 and L2 regularization, both can relieve the influence of highly correlated variables. A random-forest-like logistic regression method is proposed, random lasso [15]. It mainly consists of two steps. First, lasso method is applied to bootstrap samples. After the first step, another term, importance is added to high-weighted variables. It can select all highly correlated variables, instead of normal lasso method which only select one of the highly correlated variables.

Support vector machines (SVMs) have already been widely used in analysis microarray data [16]–[18]. It does not require independent variables and yields very good performance. But SVM cannot directly select the most important genes for classification. The decision tree method has been used to analyze microarray data. One study indicates that decision trees can handle high-dimension data and find appropriate splitting attributes for classification of cancer [19]. However, if the cancer is caused by many genes, the decision tree may not perform well because decision trees cannot readily handle interactions between variables.

5. CONCLUSION

In this study, two genes, PENK and TRAF6, are selected by all three of our basic feature selection methods. PENK encodes a signal protein which is responsible for signal transduction in the synapse between two neurons [20] and modulates the perception of pain [21], though it has not been fully studied in other types of cells. It may also work as a signal protein between adjacent RPE cells or between

RPE cells and retina which belong to neuron cells. Also, it is regulated by two crucial genes Fos and Jun [22], which regulate cell proliferation. TRAF6 is a member of the TNF receptor associated factor (TRAF) protein family. It participates in signal transduction of both the TNF receptor and the interleukin-1 receptor [23]. Through this pathway, it regulates many signaling cascades in adaptive immunity, innate immunity and bone homeostasis. It also functions as a signal transducer in the NF-kappaB pathway in response to proinflammatory cytokines and other environmental interactions [24]. It indicates that these three pathways may play an important in AMD process.

Furthermore, PCA with 12 selected genes by any two classifiers shows a surprisingly good performance, and it suggests that those genes are vital for classification. Besides, functional clustering results from DAVID is corresponding to the function of PENK and TRAF6. The clustering results suggest that those 36 genes selected by any classifier are mainly involved in signal transduction.

This paper uses data mining methods to select relevant features for classifying AMD and in so doing provides insight into the disease. AMD is completely different from other types of cancers, so the feature selection algorithms for cancers may not be useful in this case.

Finally, we cannot find one or two genes which are responsible for classification of normal and AMD eyes status, but the selected genes suggest extracellular environment does impact the process of AMD, which also agrees with the experimental studies of AMD. Furthermore, PENK and TRAF6 have not been studied for the molecular pathogenesis of AMD in biological experiments, so further biological experiments are still needed. This paper may provide new insight into finding the AMD pathological molecular pathways.

REFERENCES

[1] Cook, Briggs, et al. "Apoptotic photoreceptor degeneration in experimental retinal detachment." Investigative ophthalmology & visual science 36.6 (1995): 990-996.

[2] Klein R, Chou CF, Klein BE, Zhang X, Meuer SM, Saaddine JB: Prevalence of age-related macular degeneration in the US population. Arch Ophthalmol 2011, 129:75-80.

[3] Newman, Aaron M., et al. "Systems-level analysis of age-related macular degeneration reveals global biomarkers and phenotype-specific functional networks." Genome Med 4.2 (2012): 16.

[4] Tsai, Chen‐An, Yi‐Ju Chen, and James J. Chen. "Testing for differentially expressed genes with microarray data." Nucleic acids research 31.9 (2003): 52.

[5] Fisher RA, Yates F: Statistical Tables for Biological, Agricultural and Medical Research. Oliver and Boyd, London, England: Hafner Press; 1948.

[6] Abeel, Thomas, et al. "Robust biomarker identification for cancer diagnosis with ensemble feature selection methods." Bioinformatics 26.3 (2010): 392-398.

[7] Saeys, Yvan, Thomas Abeel, and Yves Van de Peer. "Robust feature selection using ensemble feature selection techniques." Machine learning and knowledge discovery in databases. Springer Berlin Heidelberg, 2008. 313-325.

[8] Dietterich, Thomas G. "Ensemble methods in machine learning." Multiple classifier systems. Springer Berlin Heidelberg, 2000. 1-15.

[9] Jiao, Xiaoli, et al. "DAVID-WS: a stateful web service to facilitate gene/protein list analysis." Bioinformatics 28.13 (2012): 1805-1806.

[10] Yeung, Ka Yee, and Walter L. Ruzzo. "Principal component analysis for clustering gene expression data." Bioinformatics 17.9 (2001): 763-774.

[11] Wang, Antai, and Edmund A. Gehan. "Gene selection for microarray data analysis using principal component analysis." Statistics in medicine 24.13 (2005): 2069-2087.

[12] Demichelis, Francesca, et al. "A hierarchical Naive Bayes Model for handling sample heterogeneity in classification problems: an application to tissue microarrays." BMC bioinformatics 7.1 (2006): 514.

[13] Zou, Hui. "The adaptive lasso and its oracle properties." Journal of the American statistical association 101.476 (2006): 1418-1429.

[14] Zou, Hui, and Trevor Hastie. "Regularization and variable selection via the elastic net." Journal of the Royal Statistical Society: Series B (Statistical Methodology) 67.2 (2005): 301-320.

[15] Wang, Sijian, et al. "Random lasso." The annals of applied statistics 5.1 (2011): 468.

[16] Brown, Michael PS, et al. "Knowledge-based analysis of microarray gene expression data by using support vector machines." Proceedings of the National Academy of Sciences 97.1 (2000): 262-267.

[17] Furey, Terrence S., et al. "Support vector machine classification and validation of cancer tissue samples using microarray expression data." Bioinformatics 16.10 (2000): 906-914.

[18] Guyon, Isabelle, et al. "Gene selection for cancer classification using support vector machines." Machine learning 46.1-3 (2002): 389-422.

[19] Horng, Jorng-Tzong, et al. "An expert system to classify microarray gene expression data using gene selection by decision tree." Expert Systems with Applications 36.5 (2009): 9072-9081.

[20] Comb, Michael, et al. "Proteins bound at adjacent DNA elements act synergistically to regulate human proenkephalin cAMP inducible transcription." The EMBO journal 7.12 (1988): 3793.

[21] König, Monika, et al. "Pain responses, anxiety and aggression in mice deficient in pre-proenkephalin." (1996): 535-538.

[22] Sonnenberg, June L., et al. "Regulation of proenkephalin by Fos and Jun." Science 246.4937 (1989): 1622-1625.

[23] Ye, Hong, et al. "Distinct molecular mechanism for initiating TRAF6 signalling." Nature 418.6896 (2002): 443-447.

[24] Deng, Li, et al. "Activation of the IκB kinase complex by TRAF6 requires a dimeric ubiquitin-conjugating enzyme complex and a unique polyubiquitin chain." Cell 103.2 (2000): 351-361.

ISBN: 1-60132-431-6, CSREA Press ©

SESSION

SEGMENTATION, CLUSTERING, ASSOCIATION + WEB / TEXT / MULTIMEDIA MINING + SOFTWARE

Chair(s)

Diego Galar
Peter Geczy
Robert Stahlbock
Gary M. Weiss

ISBN: 1-60132-431-6, CSREA Press ©

ISBN: 1-60132-431-6, CSREA Press ©

String Vector based AHC as Approach to Word Clustering

Taeho Jo
Department of Computer and Information Communication Engineering, Hongik University, Sejong, South Korea

Abstract— *In this research, we propose the string vector based AHC (Agglomerative Hierarchical Clustering) algorithm as the approach to the word clustering. In the previous works on text clustering, it was successful to encode texts into string vectors by improving the performance of text clustering; it provided the motivation of doing this research. In this research, we encode words into string vectors, define the semantic operation on string vectors, and modify the AHC algorithm into its string vector based version. As the benefits from this research, we expect the improved performance and more compact representations of words. Hence, the goal of this research is to implement the word clustering system with the benefits.*

Keywords: Word Clustering, String Vector, AHC Algorithm

1. Introduction

Word clustering refers to the process of segmenting a group of various words into subgroups of content based similar words. In the task, a group of arbitrary words is given as the input, and they are encoded into their structured forms. The similarity measure between the structured forms which represent the words is defined and the similarities among them are computed. The words are arranged into subgroups based on their similarities. In this research, we assume that the unsupervised learning algorithms are used for the task, although other types of approaches exist.

Let us mention the challenges which this research attempts to tackle with. In encoding words into numerical vectors for using the traditional clustering algorithms, we need many features for the robust clustering, since each feature has very weak coverage[1]. Each numerical vector which represents a word or a text tends to have the sparse distribution where zero values are dominant with more than 90%[5][9]. In previous works, we proposed that texts or words should be encoded into tables as alternative representations to numerical vectors, but it is very expensive to compute the similarity between tables[5][9]. Hence, in this research, we solve the problems by encoding words into string vectors.

Let us mention what is proposed in this research as its idea. We encode words into string vectors where elements are text identifiers which are given as symbols or codes as alternative representations to numerical vectors. We define the operation on string vectors which corresponds to the cosine similarity between numerical vectors as the similarity measure between them. We modify the AHC (Agglomerative Hierarchical Clustering) algorithm into the version where input data are given as string vectors. Hence, in this research, the words are clustered by the modified version of AHC algorithm.

Let us consider the benefits which are expected from this research. The string vectors become the more compact representations of words than numerical vectors; it requires much less features in encoding words. We expect the improved discriminations among string vectors since there is very few sparse distributions in string vectors. We expect the improved clustering performance by solving the problems which are caused by encoding words into numerical vectors. Therefore, the goal of this research is to implement the word clustering systems which are improved by the benefits.

This article is organized into the four sections. In Section 2, we survey the relevant previous works. In Section 3, we describe in detail what we propose in this research. In Section 4, we mention the remaining tasks for doing the further research.

2. Previous Works

Let us survey the previous cases of encoding texts into structured forms for using the machine learning algorithms to text mining tasks. The three main problems, huge dimensionality, sparse distribution, and poor transparency, have existed inherently in encoding them into numerical vectors. In previous works, various schemes of preprocessing texts have been proposed, in order to solve the problems. In this survey, we focus on the process of encoding texts into alternative structured forms to numerical vectors. In other words, this section is intended to explore previous works on solutions to the problems.

Let us mention the popularity of encoding texts into numerical vectors, and the proposal and the application of string kernels as the solution to the above problems. In 2002, Sebastiani presented the numerical vectors are the standard representations of texts in applying the machine learning algorithms to the text classifications [1]. In 2002, Lodhi et al. proposed the string kernel as a kernel function of raw texts in using the SVM (Support Vector Machine) to the text classification [2]. In 2004, Lesile et al. used the version of SVM which proposed by Lodhi et al. to the protein classification [3]. In 2004, Kate and Mooney used also the SVM version for classifying sentences by their meanings [4].

It was proposed that texts are encoded into tables instead of numerical vectors, as the solutions to the above problems. In 2008, Jo and Cho proposed the table matching algorithm as the approach to text classification [5]. In 2008, Jo applied also his proposed approach to the text clustering, as well as the text categorization [9]. In 2011, Jo described as the technique of automatic text classification in his patent document [7]. In 2015, Jo improved the table matching algorithm into its more stable version [8].

Previously, it was proposed that texts should be encoded into string vectors as other structured forms. In 2008, Jo modified the k means algorithm into the version which processes string vectors as the approach to the text clustering[9]. In 2010, Jo modified the two supervised learning algorithms, the KNN and the SVM, into the version as the improved approaches to the text classification [10]. In 2010, Jo proposed the unsupervised neural networks, called Neural Text Self Organizer, which receives the string vector as its input data [11]. In 2010, Jo applied the supervised neural networks, called Neural Text Categorizer, which gets a string vector as its input, as the approach to the text classification [12].

The above previous works proposed the string kernel as the kernel function of raw texts in the SVM, and tables and string vectors as representations of texts, in order to solve the problems. Because the string kernel takes very much computation time for computing their values, it was used for processing short strings or sentences rather than texts. In the previous works on encoding texts into tables, only table matching algorithm was proposed; there is no attempt to modify the machine algorithms into their table based version. In the previous works on encoding texts into string vectors, only frequency was considered for defining features of string vectors. In this research, based on [10], we consider the grammatical and posting relations between words and texts as well as the frequencies for defining the features of string vectors, and encode words into string vectors in this research.

3. Proposed Approach

This section is concerned with encoding words into string vectors, modifying the AHC (Agglomerative Hierarchical Clustering) algorithm into string vector based version and applying it to the word clustering, and consists of the three sections. In Section 3.1, we deal with the process of encoding words into string vectors. In Section 3.2, we describe formally the similarity matrix and the semantic operation on string vectors. In Section 3.3, we do the string vector based AHC version as the approach to the word clustering. Therefore, this article is intended to describe the proposed AHC version as the word clustering tool.

3.1 Word Encoding

This section is concerned with the process of encoding words into string vectors. The three steps are involved in doing so, as illustrated in Figure 1. A single word is given as the input, and a string vector which consists of text identifiers is generated as the output. We need to prepare a corpus which is a collection of texts for encoding words. Therefore, in this section, we will describe each step of encoding the words.

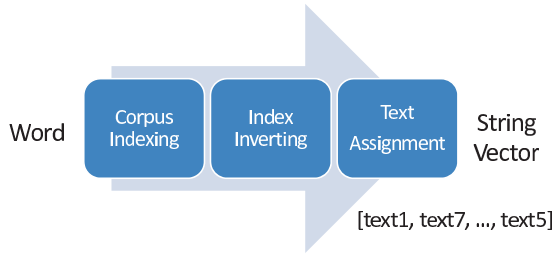


Fig. 1: Overall Process of Word Encoding

The first step of encoding words into string vectors is to index the corpus into a list of words. The texts in the corpus are concatenated into a single long string and it is tokenized into a list of tokens. Each token is transformed into its root form, using stemming rules. Among them, the stop words which are grammatical words such as propositions, conjunctions, and pronouns, irrelevant to text contents are removed for more efficiency. From the step, verbs, nouns, and adjectives are usually generated as the output.

The inverted list where each word is linked to the list of texts which include it is illustrated in Figure 2. A list of words is generated from a text collection by indexing each text. For each word, by retrieving texts which include it, the inverted list is constructed. A text and a word are associated with each other by a weight value as the relationship between them. The links of each word with a list of texts is opposite to those of each text with a list of words becomes the reason of call the list which is presented in Figure 2, inverted list.

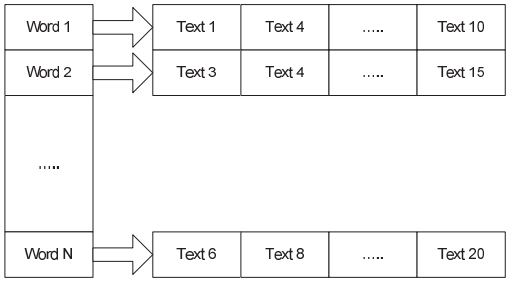


Fig. 2: The Inverted Index

Each word is represented into a string vector based on the inverted index which is shown in Figure 3. In this research, we define the features which are relations between texts and words as follows:

- Text identifier which has its highest frequency among the text collection

- Text identifier which has its highest TF-IDF weight among the text collection
- Text identifier which has its second highest frequency among the text collection
- Text identifier which has its second highest TF-IDF weight among the text collection
- Text identifier which has its highest frequency in its first paragraph among text collection
- Text identifier which has its highest frequency in its last paragraph among text collection
- Text identifier which has its highest TF-IDF weight in its first paragraph among text collection
- Text identifier which has its highest TF-IDF weight in its last paragraph among text collection

We assume that each word is linked with texts including their own information: its frequencies and its weights in the linked texts and their first and last paragraphs. From the inverted index, we assign the corresponding values which are given as text identifiers to each feature. Therefore, the word is encoded into an eight dimensional string vector which consists of eight strings which indicate text identifiers.

Let us consider the differences between the word encoding and the text encoding. Elements of each string vector which represents a word are text identifiers, whereas those of one which represents a text are word. The process of encoding texts involves the link of each text to a list of words, where as that of doing words does the link of each word to a list of texts. For performing semantic similarity between string vectors, in text processing, the word similarity matrix is used as the basis, while in word processing, the text similarity matrix is used. The relations between words and texts are defined as features of strings in encoding texts and words.

3.2 String Vectors

This section is concerned with the operation on string vectors and the basis for carrying out it. It consists of two subsections and assumes that a corpus is required for performing the operation. In Section 3.2.1, we describe the process of constructing the similarity matrix from a corpus. In Section 3.2.2, we define the string vector formally and characterize the operation mathematically. Therefore, this section is intended to describe the similarity matrix and the operation on string vectors.

3.2.1 Similarity Matrix

This subsection is concerned with the similarity matrix as the basis for performing the semantic operation on string vectors. Each row and column of the similarity matrix corresponds to a text in the corpus. The similarities of all possible pairs of texts are given as normalized values between zero and one. The similarity matrix which we construct from the corpus is the $N \times N$ square matrix with symmetry elements and 1's diagonal elements. In this subsection, we will describe formally the definition and characterization of the similarity matrix.

Each entry of the similarity matrix indicates a similarity between two corresponding texts. The two documents,d_i and d_j, are indexed into two sets of words, D_i and D_j. The similarity between the two texts is computed by equation (1),

$$sim(d_i, d_j) = \frac{2|D_i \cap D_j|}{|D_i| + |D_j|} \tag{1}$$

where $|D_i|$ is the cardinality of the set, D_i. The similarity is always given as a normalized value between zero and one; if two documents are exactly same to each other, the similarity becomes 1.0 as follows:

$$sim(d_i, d_j) = \frac{2|D_i \cap D_i|}{|D_i| + |D_i|} = 1.0$$

and if two documents have no shared words, $D_i \cap D_j = \emptyset$ the similarity becomes 0.0 as follows:

$$sim(d_i, d_j) = \frac{2|D_i \cap D_j|}{|D_i| + |D_j|} = 0.0$$

The more advanced schemes of computing the similarity will be considered in next research.

From the text collection, we build $N \times N$ square matrix as follows:

$$S = \begin{pmatrix} s_{11} & s_{12} & \dots & s_{1d} \\ s_{21} & s_{22} & \dots & s_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ s_{d1} & s_{d2} & \dots & s_{dd} \end{pmatrix}.$$

N individual texts which are contained in the collection correspond to the rows and columns of the matrix. The entry, s_{ij} is computed by equation (1) as follows:

$$s_{ij} = sim(d_i, d_j)$$

The overestimation or underestimation by text lengths are prevented by the denominator in equation (1). To the number of texts, N, it costs quadratic complexity, $O(N^2)$, to build the above matrix.

Let us characterize the above similarity matrix, mathematically. Because each column and row corresponds to its same text in the diagonal positions of the matrix, the diagonal elements are always given 1.0 by equation (1). In the off-diagonal positions of the matrix, the values are always given as normalized ones between zero and one, because of $0 \leq 2|D_i \cap D_i| \leq |D_i| + |D_j|$ from equation (1). It is proved that the similarity matrix is symmetry, as follows:

$$s_{ij} = sim(d_i, d_j) = \frac{2|D_i \cap D_j|}{|D_i| + |D_j|} = \frac{2|D_j \cap D_i|}{|D_j| + |D_i|}$$
$$= sim(d_j, d_i) = s_{ji}$$

Therefore, the matrix is characterized as the symmetry matrix which consists of the normalized values between zero and one.

The similarity matrix may be constructed automatically from a corpus. The N texts which are contained in the corpus are given as the input and each of them is indexed into a list of words. All possible pairs of texts are generated and the similarities among them are computed by equation (1). By computing them, we construct the square matrix which consists of the similarities. Once making the similarity matrix, it will be used continually as the basis for performing the operation on string vectors.

3.2.2 String Vector and Semantic Similarity

This section is concerned with the string vectors and the operation on them. A string vector consists of strings as its elements, instead of numerical values. The operation on string vectors which we define in this subsection corresponds to the cosine similarity between numerical vectors. Afterward, we characterize the operation mathematically. Therefore, in this section, we define formally the semantic similarity as the semantic operation on string vectors.

The string vector is defined as a finite ordered set of strings as follows:

$$\mathbf{str} = [str_1, str_2, ..., str_d]$$

An element in the vector, str_i indicates a text identifier which corresponds to its attribute. The number of elements of the string vector, $\mathbf{str}$ is called its dimension. In order to perform the operation on string vectors, we need to define the similarity matrix which was described in Section 3.2.1, in advance. Therefore, a string vector consists of strings, while a numerical vector does of numerical values.

We need to define the semantic operation which is called 'semantic similarity' in this research, on string vectors; it corresponds to the cosine similarity on numerical vectors. We note the two string vectors as follows:

$$\mathbf{str}_1 = [str_{11}, str_{12}, ..., str_{1d}]$$

$$\mathbf{str}_2 = [str_{21}, str_{22}, ..., str_{2d}]$$

where each element, d_{1i} and d_{21i} indicates a text identifier. The operation is defined as equation (3.2.2) as follows:

$$sim(\mathbf{str}_1, \mathbf{str}_2) = \frac{1}{d}\sum_{i=1}^{d} sim(d_{1i}, d_{2i}) \qquad (2)$$

The similarity matrix was constructed by the scheme which is described in Section 3.2.1, and the $sim(d_{1i}, d_{2i})$ is computed by looking up it in the similarity matrix. Instead of building the similarity matrix, we may compute the similarity, interactively.

The semantic similarity measure between string vectors may be characterized mathematically. The commutative law applies as follows:

$$sim(\mathbf{str}_1, \mathbf{str}_2) = \frac{1}{d}\sum_{i=1}^{d} sim(d_{1i}, d_{2i})$$

$$= \frac{1}{d}\sum_{i=1}^{k} sim(d_{2i}, d_{1i}) = sim(\mathbf{str}_2, \mathbf{str}_1)$$

If the two string vectors are exactly same, its similarity becomes 1.0 as follows:

$$\text{if } \mathbf{str}_1 = \mathbf{str}_2 \text{with } \forall_i sim(d_{1i}, d_{2i}) = 1.0$$

$$\text{then } sim(\mathbf{str}_1, \mathbf{str}_2) = \frac{1}{d}\sum_{i=1}^{d} sim(d_{1i}, d_{2i}) = \frac{d}{d} = 1.0$$

However, note that the transitive rule does not apply as follows:

$$\text{if } sim(\mathbf{str}_1, \mathbf{str}_2) = 0.0 \text{ and } sim(\mathbf{str}_2, \mathbf{str}_3) = 0.0$$

$$\text{then, not always } sim(\mathbf{str}_1, \mathbf{str}_3) = 0.0$$

We need to define the more advanced semantic operations on string vectors for modifying other machine learning algorithms. We define the update rules of weights vectors which are given as string vectors for modifying the neural networks into their string vector based versions. We develop the operations which correspond to computing mean vectors over numerical vectors, for modifying the k means algorithms. We consider the scheme of selecting representative vector among string vectors for modifying the k medoid algorithms so. We will cover the modification of other machine learning algorithms in subsequent researches.

3.3 The Proposed Version of AHC Algorithm

This section is concerned with the proposed AHC version as the approach to the text clustering. Raw texts are encoded into string vectors by the process which was described in Section 3.2.1. In this section, we attempt to the traditional AHC into the version where a string vector is given as the input data. The version is intended to improve the clustering performance by avoiding problems from encoding texts into numerical vectors. Therefore, in this section, we describe the proposed AHC version in detail, together with the traditional version.

The traditional version of AHC algorithm is illustrated in Figure 3. Words are encoded into numerical vectors, and it begins with unit clusters each of which has only single item. The similarity of every pairs of clusters is computed using the Euclidean distance or the cosine similarity, and the pair with its maximum similarity is merged into a cluster. The clustering by the ACH algorithm proceeds by merging cluster pairs and decrementing number of clusters by one. If the similarities among the sparse numerical vectors are computed, the traditional version becomes very fragile from the poor discriminations among them.

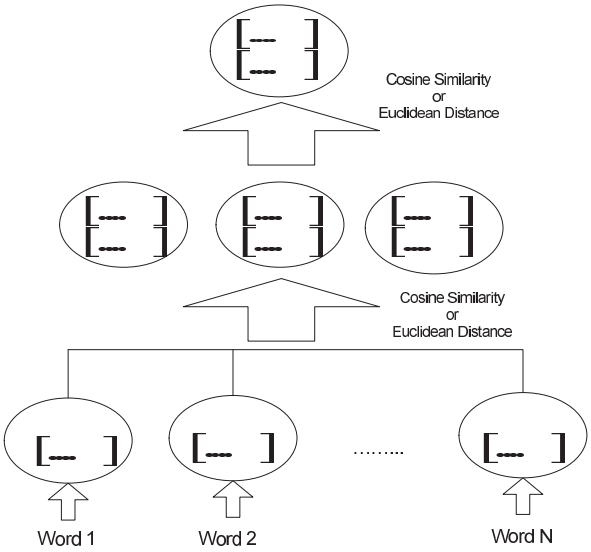


Fig. 3: The Traditional Version of AHC Algorithm

Separately from the traditional version, the clustering process by the proposed AHC version is illustrated in Figure 4. Texts are encoded into string vectors, and the algorithm begins with unit clusters each of which has a single one. The similarities of all possible pairs of clusters are computed and the pair with its maximum similarity is merged into a single cluster. The clustering proceeds by iterating the process of computing the similarities and merging a pair. Because the sparse distribution in each string vector is never available inherently, the poor discriminations by sparse distributions are certainly overcome in this research.

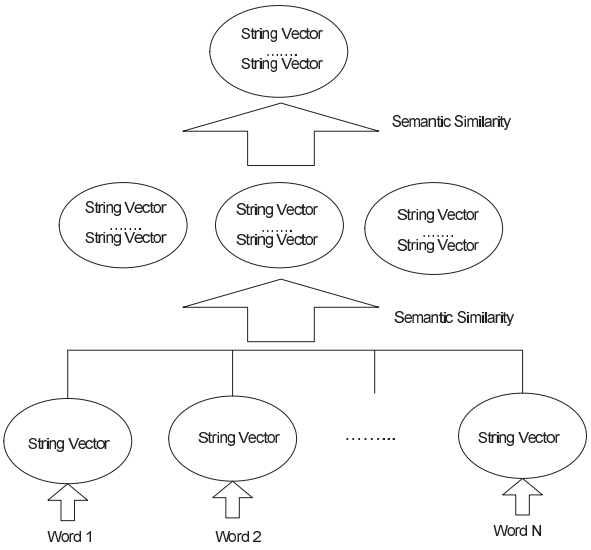


Fig. 4: The Proposed Version of AHC Algorithm

We may consider several schemes of computing a similarity between clusters. We may compute similarities of all possible pairs of items between two clusters and average over them as the cluster similarity. The maximum or the minimum among similarities of all possible pairs is set as the cluster similarity. In another scheme, we may select representative members of two clusters and the similarity between the selected members is regarded as the cluster similarity. In this research, we adopt the first scheme for computing the similarity between two clusters in using the AHC algorithm; other schemes will be considered in next research.

Because string vectors are characterized more symbolically than numerical vectors, it is easy to trace results from clustering items in the proposed version. It is assumed that the clustering proceeds by merging pairs of clusters or data items by their similarities as shown in Figure 3 and 4. The similarity between string vectors is computed by the scheme which is described in Section 3.2.2. A particular entity is nominated and its elements are extracted from it. We present the evidence of clustering entities by associating it with elements from other string vectors within the belonging clustering.

4. Conclusion

Let us mention the remaining tasks for doing the further research. The proposed approach should be validated and specialized in the specific domains: medicine, engineering and economics. Other features such as grammatical and posting features may be considered for encoding words into string vectors as well as text identifiers. Other machine learning algorithms as well as the AHC may be modified into their string vector based versions. By adopting the proposed version of the AHC, we may implement the word clustering system as a real program.

5. Acknowledgement

This work was supported by 2016 Hongik University Research Fund.

References

[1] F. Sebastiani, “Machine Learning in Automated Text Categorization", pp1-47, ACM Computing Survey, Vol 34, No 1, 2002.
[2] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins, “Text Classification with String Kernels", pp419-444, Journal of Machine Learning Research, Vol 2, No 2, 2002.
[3] C. S. Leslie, E. Eskin, A. Cohen, J. Weston, and W. S. Noble, “Mismatch String Kernels for Discriminative Protein Classification", pp467-476, Bioinformatics, Vol 20, No 4, 2004.
[4] R. J. Kate and R. J. Mooney, “Using String Kernels for Learning Semantic Parsers", pp913-920, Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics, 2006.
[5] T. Jo and D. Cho, “Index based Approach for Text Categorization",International Journal of Mathematics and Computers in Simulation, Vol 2, No 1, 2008.
[6] T. Jo, “Single Pass Algorithm for Text Clustering by Encoding Documents into Tables", pp1749-1757, Journal of Korea Multimedia Society, Vol 11, No 12, 2008.

[7] T. Jo, "Device and Method for Categorizing Electronic Document Automatically", Patent Document, 10-2009-0041272, 10-1071495, 2011.
[8] T. Jo, "Normalized Table Matching Algorithm as Approach to Text Categorization", pp839-849, Soft Computing, Vol 19, No 4, 2015.
[9] T. Jo, "Inverted Index based Modified Version of K-Means Algorithm for Text Clustering", pp67-76, Journal of Information Processing Systems, Vol 4, No 2, 2008.
[10] T. Jo, "Representationof Texts into String Vectors for Text Categorization", pp110-127, Journal of Computing Science and Engineering, Vol 4, No 2, 2010.
[11] T. Jo, "NTSO (Neural Text Self Organizer): A New Neural Network for Text Clustering", pp31-43, Journal of Network Technology, Vol 1, No 1, 2010.
[12] T. Jo, "NTC (Neural Text Categorizer): Neural Network for Text Categorization", pp83-96, International Journal of Information Studies, Vol 2, No 2, 2010.

Integrating Sequential Pattern Mining Techniques and Support Vector Machines for Sequence Classification

Chieh-Yuan Tsai and Yu-Yu Yao

Abstract—**Sequence classification problem can be found in many real world applications such as protein function prediction, text classification, and so on. Support Vector Machines (SVMs) have been wildly used to deal with sequence classification problem, since SVMs can deal with the nonlinear data and possess high efficiency in classification. However, the most difficult part in SVMs is to design an appropriate kernel function. Therefore, a pairwise sequence similarity kernel is proposed which takes sequential patterns instead of taking k-mers as reference sequences and evaluates the similarity scores between reference sequences and sequence data by the proposed map function. To obtain sequential patterns, three different sequential pattern mining methods are used to extract frequent sequential patterns, frequent closed sequential patterns, and frequent maximal sequential patterns from sequence databases. The three sequential patterns are then evaluated to know which one could achieve higher classification accuracy. A map function, which is similar to edit distance concept, is used in the proposed kernel to calculate the similarity score. Next, the sequence SVM classifier is built according to the proposed pairwise sequence similarity kernel. A set of artificial datasets are employed to test the proposed SVM classification model. The experiment results indicate the proposed SVM classification model using pairwise sequence similarity kernel is efficient and accurate.**

Keywords—sequential pattern mining; sequence classification; kernel method; SVM classifiers

I. INTRODUCTION

A plenty of classifiers such as Neural Network (NN), Naïve Bayes classification, Decision Tree (DT), and Support Vector Machine (SVM) have been proposed. Among them, SVM is one of the most popular methods. Originally, SVM was developed from Vapnik-Chervonenkis (VC) theory and structural risk minimization (SRM) principle [1, 2]. In order to get the best generalization ability and keep resistant to over fitting, SVM attempts to seek out two things. One is to minimize the training set error, and another one is to maximize the margin. Moreover, SVM can provide global minimum and avoid being trapped in local minimum due to the use of convex quadratic programming. SVMs can deal with the nonlinear data and possess high efficiency in the process of classification.

This work was partially supported by the National Science Council of Taiwan (No. 102-2221-E-155-041-MY3).

Chieh-Yuan Tsai is with Department of Industrial Engineering and Management, and Innovation Center for Big Data and Digital Convergence, Yuan Ze University, Taoyuan City, Taiwan (corresponding author: +886-3-4638800 Ext. 2512; fax: +886-3-463-8907; e-mail: cytsai@saturn.yzu.edu.tw).

Yu-Yu Yao, was with Department of Industrial Engineering and Management, Yuan Ze University, Taoyuan City, Taiwan. He is now with Hon-Hai Precision Industry Co.

In SVM, the selection of an appropriate kernel function significantly affect the performance of its operation [3, 4]. Kernel functions enable them to operate in a high-dimensional implicit feature space without ever computing the coordinates of the data in that space. When dealing with numerical data, polynomial kernel and radial-basis function (RBF) kernel are most popular ones. However, many applications , such as text categorization, gene organization rules, and protein fold recognition, are not numerical data but sequence data. This make some popular kernels unsuitable for sequence applications.

[5] proposed Fisher kernel for discrete symbol sequences using a global discrete hidden Markov model (DHMM). [6, 7] presented the spectrum kernel and mismatch kernel where these two kernels used the number of common k-mers as reference sequences. [8] developed the string subsequence kernel that maps a string into a feature vector whose dimension is equal to the number of all possible subsequences of a particular length. This kernel, similar to spectrum kernel and mismatch kernel, considers two sequences as very similar if they share many common substrings. [9, 10, 11] considered the frequency log likelihood ratio based on the probabilities of occurrence of k-mers to be reference sequences. The term frequency log likelihood ratio kernel, which is similar to spectrum kernel and subsequence kernel, maps a discrete symbol sequence onto a feature space.

Those previously mentioned kernels are based on k-mers because k-mers provide a simple way to construct the reference sequence. However, when the number of symbols is large, the number of dimensional feature space increased significantly. For example, if the number of symbols is 12 and k is 3, the number of dimensional feature space is 12^3=1728. Among 1728-dimensional feature space, many worthless dimensional feature spaces are involved. To solve this problem, this paper proposes a method that takes frequent sequential patterns as reference sequences so that the computing cost can be significantly reduced. Specifically, this research develops an efficient support vector machine classifier for sequence data. The proposed sequence classification method consists of two parts: sequential pattern mining and sequence SVM classifier. In sequential pattern mining part, this research provides efficient and useful algorithms to generate three sequential patterns, which are frequent sequential patterns, frequent closed sequential patterns, and frequent maximal sequential patterns, are derived as the reference sequences. The main reason is the amounts of the three sequential patterns might be very different from mining processes. In sequence SVM classifier, the pairwise sequence similarity (PSS) kernel is developed. Map function, which is edit distance algorithm, is

used to compute the similarity scores between sequences data and reference sequences in database. Finally, the sequence data is transformed as a pairwise sequence similarity score vector as input to train the sequence SVM classifier by the proposed kernel method.

II. Sequential Pattern Mining

Sequential pattern mining algorithms address the problem of discovering the existent maximal frequent sequences in a given database. The problem was first introduced by Agrawal and Srikant, where the basic concepts involved in pattern detection were established [12]. In the recent years, several sequential pattern mining algorithms were proposed, but not all assume the same conditions. Some basic definitions are needed, in order to formally introduce the problem. An itemset is a non-empty subset of elements from the item collection, called items. If the data is time dependent, an itemset corresponds to the set of items transacted in a particular instant by a particular entity. The itemset composed of items a and b is denoted by (ab). A sequence is an ordered list of itemsets. A sequence is maximal if it is not contained in any other sequence. A sequence with k items is called a k-sequence. The number of elements (itemsets) in a sequence s is the length of the sequence and is denoted by |s|. The *i*th itemset in the sequence is represented by s_i. and < > denotes the empty sequence. The result of the concatenation of two sequences x and y is a new sequence denoted by xy. The set of considered sequences is usually designated by database (DB), and the number of sequences by database size |DB|. A sequence $a=<a_1a_2...a_n>$ is contained in another sequence $b=<b_1b_2...b_m>$, or a is a subsequence of *b*, if there exist integers $i_1<i_2<\ldots<i_n$ such that $a_1 \subseteq b_{i1}$, $a_2 \subseteq b_{i2}$, …, $a_n \subseteq b_{in}$.

A subsequence s' of s is denoted by $s' \subseteq s$, and by $s' \subset s$ if s' is a proper subsequence of s, i.e. if s' is a subsequence of s but is not equal to *s*. It is usual to assume that the items in an itemset of a sequence are in lexicographic order. This assumption facilitates the design of sequential pattern mining algorithms, avoiding the repetition of some operations (such as the generation of repeated sequences). In this manner, prefixes and suffixes have specific meanings. They are special cases of subsequences: the sequence without the first element in the first itemset of the sequence and without the last element of the last itemset of the sequence, respectively. Finally, the sequential pattern-mining problem may be stated in its entirety. Given a database D of sequences, and some user-specified minimum support threshold σ and constraint c, a sequence is frequent if it is contained in at least σ sequences in the database, satisfying the constraint c.

III. The Proposed Method

The framework of proposed sequence classification method is depicted in Fig. 1. A sequence database *SD* is divided into *n* sub-databases according to the class label of each sequence. Let $SD^c = \{<S_j, c>\}$ be the *c*th sub-database where S_j represents the *j*th sequence, *c* represents the class label for sequence S_j where c=1, 2… *n*. For each sub-database SD^c, a sequential pattern mining algorithm is applied for so that a set of sequential patterns SP^c can be generated. Then, a pair of subsets $\{SP^i, SP^j\}$ and their original sub-database pair $\{SD^i, SD^j\}$ are used to build a sequence SVM classifier where $i, j \in \{1, 2\ldots n\}$, $i \neq j$. That is, there are $n(n\text{-}1)/2$ SVM classifiers in total. For example, if n = 3, three classifiers are constructed using data of $<\{SD^1, SD^2\}, \{SP^1, SP^2\}>$, $<\{SD^2, SD^3\}, \{SP^2, SP^3\}>$, and $<\{SD^1, SD^3\}, \{SP^1, SP^3\}>$ respectively. When an unknown sequence is inputted, this sequence will be classified by each classifier. Finally, a majority voting scheme is applied based on the class decision of each SVM classifier.

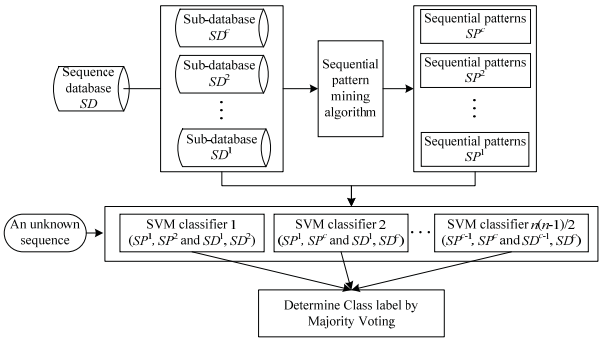


Fig. 1. Framework of the proposed sequence classification method.

A. Sequential pattern mining algorithms

In this study, three sequential patterns are extracted from the same sub-database. They are frequent sequential patterns, frequent closed sequential patterns, and frequent maximal sequential patterns. This research applies PrefixSpan algorithm [13] for generating frequent sequential patterns, CloSpan algorithm [14] for generating frequent closed sequential patterns, and VMSP algorithm [15] for frequent maximal sequential patterns. The three sequential patterns are then evaluated later to know which one could achieve higher classification accuracy.

To make the following discussion easier, a set of frequent sequential patterns is denoted as $FSP^c = \{ fsp_1^c, fsp_2^c, \ldots, fsp_{M_c}^c \}$, a set of frequent closed sequential patterns is denoted as $FCSP^c = \{ fcsp_1^c, fcsp_2^c, \ldots, fcsp_{N_c}^c \}$, and a set of frequent maximal sequential patterns is denoted as $FMSP^c = \{ fmsp_1^c, fmsp_2^c, \ldots, fmsp_{O_c}^c \}$.

B. Sequence Support Vector Machine Classifier

Fig. 2 shows the process of constructing a sequence SVM classifier. There are two main parts in this classifier. The first part is the pairwise sequence similarity kernel. The patterns derived previously are used as reference sequence vectors in the kernel. Therefore, there are three different reference sequence vectors *RS*. Then, all sequences in the database will conduct feature map based on the three reference sequence vectors. The map function *MF* used in this study is edit distance algorithm. The purpose of conducting feature map is to map the sequence into high dimensional feature space. After mapping by edit distance algorithm, pairwise sequence similarity score vectors are obtained.

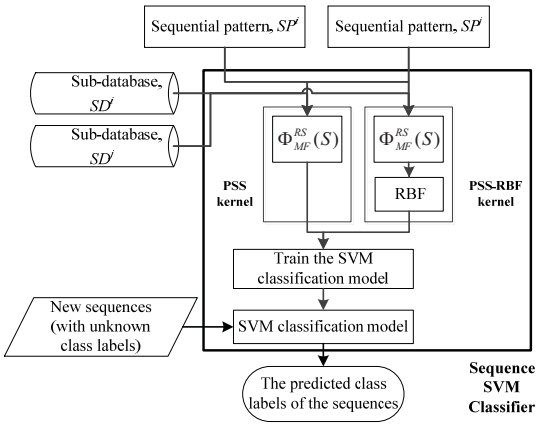


Fig. 2. The proposed sequence SVM classifier with pairwise sequence similarity kernel

The pair of FSP^i and FSP^j is considered as reference sequences for frequent sequential patterns. That is, the reference sequence vectors can be represented as $RS_{fsp} = < fsp_1^i, fsp_2^i, ..., fsp_{M_i}^i, fsp_1^j, fsp_2^j, fsp_{M_j}^j >$. Similarly, the reference sequence vectors for frequent closed sequential patterns and frequent maximum sequential patterns respectively can be expressed as $RS_{fcsp} = < fcsp_1^i, fcsp_2^i, ..., fcsp_{N_i}^i, fcsp_1^j, fcsp_2^j, fcsp_{N_j}^j >$, and $RS_{fmsp} = < fmsp_1^i, fmsp_2^i, ..., fmsp_{O_i}^i, fmsp_1^j, fmsp_2^j, fmsp_{O_j}^j >$. To make the following explanation easier, the above reference sequence vector is formulated as:

$$RS = < rs_1, rs_2, \cdots, rs_w > \quad (1)$$

where rs_x is one of reference sequences.

In this study, a map function which edit distance algorithm is applied for, is denoted as *MF*. The definition of edit distance is the minimal penalty cost between a sequence, *S*, and a reference sequence, rs_x, assessed by sequence similarity. The similarity score of edit distance between two sequences are computed, and an edit distance required to transform rs_x to sequence *S*. Four operations of edit distance which are "substitution", "insertion", "deletion", and "no change". One itemset of rs_x is replaced by the corresponding element in *S* is called "substitution." "Insertion" is that one element of sequence *S* is inserted into one rs_x. Thus, the length of rs_x will increase by one element. One element of sequence *S* is deleted is called "deletion." Therefore, the length of rs_x decreases by one element. "No change" is that itemset of rs_x is the same as the corresponding element in *S*.

Let $\varphi_{MF}^{RS}(S, rs_x)$ be the similarity score function that evaluates the score between sequence *S* and rs_x. The pairwise sequence similarity score vector for sequence *S* is given by

$$\Phi_{MF}^{RS}(S) = < \varphi_{MF}^{RS}(S, rs_1), \varphi_{MF}^{RS}(S, rs_2), ..., \varphi_{MF}^{RS}(S, rs_W) > \quad (2)$$

Alternatively, the pairwise sequence similarity score vector for sequence *S* can be represented as

$$\Phi_{MF}^{RS}(S) = \left\langle \varphi_{MF}^{RS}(S, rs_x) \right\rangle_{x=1}^{w} \quad (3)$$

In addition, to know the performance of the proposed PSS kernel, another version of pairwise sequence similarity with Radial Basis Function (RBF) kernel, denoted as PSS-RFB kernel, is compared. RBF is a popular kernel function which it is commonly used in SVM classification. Two sequences *u* and *v*, represented as feature vectors, is defined as $K(u,v) = exp(-\gamma||x_i-x_j||^2)$, $\gamma>0$ where $||x_i-x_j||^2$ may be viewed as the squared Euclidean distance between the two feature vectors. γ is the free parameter that implicitly defines the non-linear mapping from input space to some high-dimensional feature space.

The second part is to construct the sequence SVM classifier. The sequence SVM classifier is built according to the proposed pairwise sequence similarity kernel, denoted as PSS kernel. Through the proposed sequence classification framework, the class label of a new sequence will be predicted precisely.

C. Support Vector Machine for Binary and Multiple Classification

The SVM were derived intrinsically for binary classification. There are two main approaches for multi-class SVM classification. One is directly considering all data in one optimization formulation, and another one is to construct binary schemes which break down the multi-class problem into a number of smaller binary problems [16]. The former that the number of variables are used to build and solve the optimization problem for nonlinear SVM, is a positive function of the number of classes [17]. Therefore, it is computationally more expensive to solve a multi-class problem with the same number of data than binary schemes. To avoid the expensive cost of computation, the binary schemes are decided and used in this multi-class problem.

The typical implementation methods of multi-class SVM are one-against-all (OAA) and one-against-one (OAO). In the OAA scheme, the number of binary classifiers is equal to the number of classes. OAA makes the binary decisions between each class and all the other classes. In the OAO scheme, the training of n(n-1)/2 binary classifiers is required, where each class separates into two opposite classes in a pairwise way. A voting scheme is used for deciding the final class in OAO approach, and the final class belongs to the one with maximum number of votes. OAO is employed in this research, because OAO is more practical. [16] observed the training process of OAO is quicker than OAA; moreover, [18] and [19] claimed OAO is more accurate than the OAA strategy. That is the main reason why this research uses the OAO approach with nonlinear SVM.

IV. Experiments

A. Datasets

The following experiments are conducted using C# programming language under the environment of Intel(R) Core(TM) i5 2300 CPU with 4.0 GB RAM. Four artificial

datasets are introduced. The number of sequence, the length of sequence, the characters of sequence and total number of classes, sequence composition are generated by a sequence generator for artificial datasets. The detail of the four artificial datasets are summarized and shown in Table I.

TABLE I. THE DETAIL OF ARTIFICIAL DATASET

Dataset	No. of sequences	Len. of sequences	Characters used in sequences	No. of class labels	Sequence composition
*D*1	300	[4, 8]	12	3	[70%,20%,10%]
*D*2	30000	[8, 12]	12	3	[70%,20%,10%]
*D*3	300	[4, 8]	12	3	[60%,20%,20%]
*D*4	30000	[8, 12]	12	3	[60%,20%,20%]

Let's take dataset *D*1 as an example. The sequence in artificial datasets is composed of itemsets. The set of item is assumed as I = {A, B, C, D, E, F, G, H, I, J, K, L}. These 12 items are grouped into three parts, each part contains three items. For instance, {A, B, C} belongs to the first part, {D, E, F} belongs to the second part, and {G, H, I} belongs to the third part. There are three classes (c = 3), and the total sequences are 300 in *D*1 case. In addition, each class has 100 sequences. For each class 80% of sequences are used for the training dataset (*TR*) and 20% of sequences are for the testing dataset (*TE*). The length of sequences is randomly assigned as the range from 4 to 8. For the sequences in class 1, 70% of items are from the first part , 20% of items are from the second part, and the remaining 10% items are from the third part. With the same idea, for the sequences in class 2, 10% of items are from the first part , 70% of items are from the second part, and the remaining 20% items are from the third part. Similarity, for the sequences in class 3, 20% of items are from the first part , 10% of items are from the second part, and the remaining 70% items are from the third part. Table II shows part of sequences of training dataset in *D*1.

TABLE II. THE TRAINING DATA IN D1

SD^c	*D*1		
	sid	S_i	c_i
SD^1	1	<A, D, B, L>	1
	2	<A, C, B, I, C, G, H>	1
	3	<A, E, B, E, C, D, B>	1
	⋮	⋮	⋮
	79	<C, D, C, A>	1
	80	<A, G, B, E, C>	1
SD^2	81	<D, A, C, E, F, A, E>	2
	82	<E, A, D, E, F, L>	2
	83	<F, B, G, H, E, F, D, C>	2
	⋮	⋮	⋮
	159	<E, F, A, D, E, L, D, G>	2
	160	<F, E, F, H, B, L, D, E>	2
SD^3	161	<G, D, K, C, H, B, H>	3
	162	<H, F, A, G, I, C, G, H>	3
	163	<I, A, F, H, G>	3
	⋮	⋮	⋮
	239	<I, B, L, G, H, E, I, H>	3
	240	<H, A, I, G>	3

B. A Case Study

First, the sequences of training data in each class are used to derive. Frequent sequential patterns for class c are generated, denoted as FSP^c where c = 1, 2, and 3 by PrefixSpan algorithm. Table III shows the set of frequent sequential patterns for each class when minimum support is set as 0.3. In addition, frequent closed sequential patterns $FCSP^c$ and frequent maximal sequential patterns $FMSP^c$ are mined using CloSpan algorithm and VMSP algorithm respectively.

TABLE III. FREQUENT SEQUENTIAL PATTERNS IN EACH CLASS

Frequent sequential patterns		
FSP^1	FSP^2	FSP^3
<A>	<D>	<G>
<A, B>	<D, E>	<G, F>
<A, B, C>	<D, F>	<G, G>
<A, B, D>	<D, E, G>	<G, H>
<A, B, D, C>	<D, G>	<H>
<A, C>	<E>	<J>
<A, D>	<E, G>	<K>
<A, D, C>	<F>	<L>
<A, E>	<A>	<A>
<B>	<B>	<B>
<B, C>	<C>	<C>
<B, D>	<G>	<D>
<B, D, C>	<G, F>	<E>
<C>	<G, H>	<F>
<D>	<H>	
<D, C>	<L>	
<D, E>		
<E>		
<G>		

Then, the process of computing the similarity measure between sequential patterns and sequences in *D*1 using edit distance algorithm is performed. Similar process is applied to the testing data in *D*1. Therefore, three similarity score vectors for sequences and frequent sequential patterns, sequences and frequent closed sequential patterns, and sequences and frequent maximal sequential patterns.

Three similarity score vectors which has been generated are used to establish three SVM classifiers. When building SVM classification models, PSS kernel without RBF (called PSS) and PSS with RBF kernel (called PSS-RBF) are evaluated and compared. The values of gamma and cost in RBF are set as 1. Table IV shows the training and testing accuracy of classification model when the value of minimum support is 0.3. For training data, the accuracy of using *FSP* and *FCSP* are better than the accuracy of using *FMSP* when either PSS kernel or PSS-RBF kernel is applied. However, for testing data, the accuracy of using *FMSP* is better than the one of using *FSP* and *FCSP*.

TABLE IV. THE ACCURACY OF THE CLASSIFICATION MODEL WHEN *MIN_SUP* IS 0.3

	PSS kernel		PSS-RBF kernel	
	Training	Testing	Training	Testing
FSP	70.833%	66.667%	97.5%	50%
FCSP	70.833%	66.667%	97.5%	46.667%
FMSP	70%	70%	97.083%	51.667%

Table V illustrates the computational time for training the classification model and calculating accuracy for different classification models. Since the number of *FSP* is largest, computational time for the model using *FSP* will be longer than the models using *FCSP* or *FMSP*. On the other hand, the number of *FMSP* is fewest so that the computational time for training the classification model and calculating accuracy is the fastest. In addition, the computational time for the three models using PSS-RBF kernel is longer than that the computational time for the three models using PSS kernel.

TABLE V. THE COMPUTATIONAL TIME WHEN *MIN_SUP* IS 0.3

	PSS kernel	PSS-RBF kernel
FSP	0.0315	0.0392
FCSP	0.0312	0.0369
FMSP	0.0222	0.0267

C. Experimental Designs

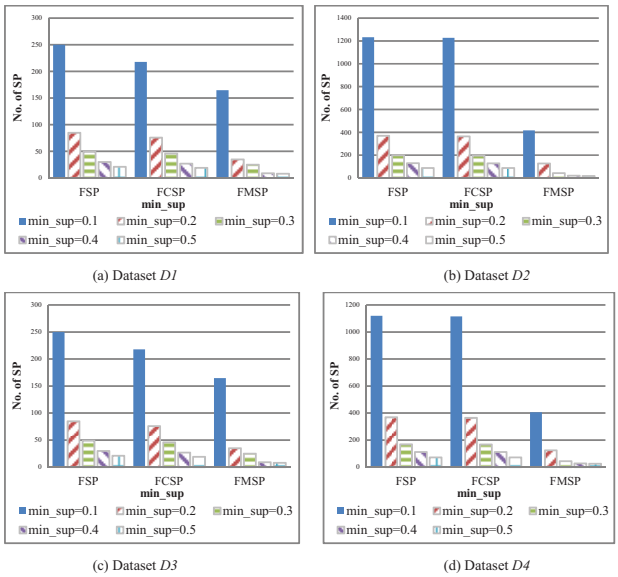
1) Numbers of Sequential Patterns

For dataset *D*1, the numbers of derived frequent sequential patterns (*FSP*), frequent closed sequential patterns (*FCSP*), and frequent maximal sequential patterns (*FMSP*) are summarized in Table VI where minimum support is adjusted from 0.1 to 0.5. *c*1, *c*2 and *c*3 in Table VI represent the number of sequential patterns in each class. It is clear that if minimum support is larger, the number of sequential patterns is less. Moreover, the number of frequent sequential patterns is larger than the numbers of other two types of sequential patterns.

TABLE VI. THE NUMBER OF SEQUENTIAL PATTERNS UNDER DIFFERENT MINIMUM SUPPORT IN *D*1

min_sup	*FSP*(*c*1,*c*2,*c*3)	*FCSP*(*c*1,*c*2,*c*3)	*FMSP*(*c*1,*c*2,*c*3)
0.1	250(82,81,87)	218(67,72,79)	165(31,62,72)
0.2	85(29,27,29)	76(25,23,28)	35(9,11,15)
0.3	49(19,16,14)	46(16,16,14)	25(4,9,12)
0.4	30(17,8,5)	27(14,8,5)	9(3,2,4)
0.5	21(12,6,3)	19(10,6,3)	8(3,3,2)

Fig. 3 shows the numbers of *FSP*, *FCSP*, *FMSP* for D1, D2, D3, and D4. In *D*2, when the value of minimum support is 0.1, the number of generated frequent sequential patterns and frequent closed sequential patterns are over 1000. On the contrary, while the value of minimum support is set as 0.5, the number of three types of sequential patterns are all less than 100. Furthermore, the number of frequent sequential patterns and frequent closed sequential patterns are identical when the value of minimum support is 0.5. In *D*4, the number of frequent sequential patterns and frequent closed sequential patterns in class 2 and class 3 are totally the same when minimum support is from 0.1 to 0.5. Moreover, when the value of minimum support is 0.4 and 0.5, the total number of frequent sequential patterns and frequent closed sequential patterns are identical. It makes the classification models build using the two sequential patterns are almost the same.



(a) Dataset *D1* (b) Dataset *D2*

(c) Dataset *D3* (d) Dataset *D4*

Fig. 3. The numbers of sequential patterns for datasets *D*1 to *D4*.

According to the above figures, the total numbers of sequential patterns in *D*1 and *D*3 are close since the numbers of sequences in *D*1 and *D*3 are all 300. Similarly, the total number of sequential patterns in *D*2 and *D*4 are close since the numbers of sequences in *D*2 and *D*4 are all 30000. This makes the total numbers of sequence patterns in *D*2 and *D*4 are greater than the ones in *D*1 and *D*3. In addition, the sequence composition in *D*1 and *D*3 are 70%, 20% and 10% and 60%, 20% and 20% respectively. Similarly, the sequence composition in *D*2 and *D*4 are 70%, 20% and 10% and 60%, 20% and 20% respectively. However, it is observed that the number of sequential patterns will not be affected by the characteristics of the sequence composition as mentioned in Table I.

2) Comparison between the Models Using PSS and PSS-RBF Kernels

Since the experiment result for *D*1 to *D*4 show the similar trend, only the comparison result of *D*1 are reported. Fig. 4 shows the comparison result when *FSP*, *FCSP* and *FMSP* are applied respectively. As shown in Fig. 4(a), for training data of *D*1, the accuracy of the model using PSS-RBF kernel is always higher than the model using PSS kernel for *FSP* case. However, for testing data of *D*1, the accuracy of model with using PSS-RBF kernel is lower than the one using PSS kernel in Fig. 4(b). The reason is that the RBF function makes the model overtrained so that the training accuracy of model is higher and the testing accuracy of model is lower. Similarly, the accuracy of model with *FCSP* and *FMSP* cases using PSS and PSS-RBF kernels are shown in Fig. 4(c) to 4(f).

(a) Training data for FSP case (b) Testing data for FSP case

(c) Training data for FCSP case (d) Testing data for FCSP case

(e) Training data for FMSP case (f) Testing data for FMSP case

Fig. 4. The testing accuracy of model for different patterns using PSS and PSS-RBF kernels under different *min_sup*

The computational time of training classification model and computing accuracy using *FSP* is shown in Fig. 5. It is clearly that when minimum support is smaller, the computational time is longer. Furthermore, the computational time of using PSS kernel is less than using PSS-RBF kernel. The figure shows that SVM classification model using PSS kernel employs less time to achieve the higher accuracy. Therefore, the SVM classification model using PSS kernel is more efficient than the one using PSS-RBF kernel. Similar findings can be found for the model using *FCSP* and *FMSP*.

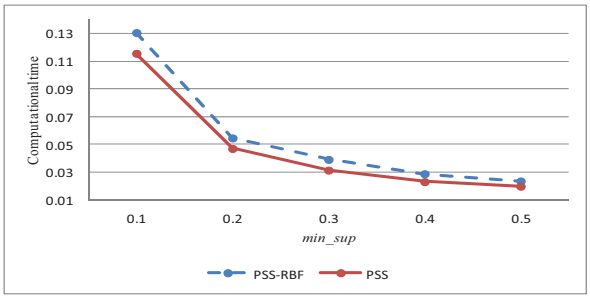


Fig. 5. The computational time of models using PSS-RBF and PSS kernels for *FSP* case

3) Comparison between the Models using PSS kernel and Spectrum Kernel

In this section, the classification model using spectrum kernel is used to compare with the model using the proposed PSS kernel. In spectrum kernel, *k*-mers where *k* is set as 2 and 3 is used as reference sequences in this study. When *k* is set as 2 (2-mers), the number of reference sequence is 12^2=144 since the number of the total items is 12. Similarly, if *k* is 3 (3-mers), the number of reference sequence is 12^3=1728.

Table VII shows the accuracy comparison for the models using PSS and spectrum kernels for *D*1. For training data, the accuracy of the proposed model is higher than the one using 2-mers and 3-mers. Similarly, the accuracy of the proposed model is higher than the one using 2-mers and 3-mers. Compared to spectrum kernel, the model using PSS can generate the highest testing accuracy and good training accuracy at the same time. When the value of minimum support is 0.1, the number of reference sequences for the model using *FSP* is 250. The number of reference sequences using PSS is more than the ones 2-mers, so the computational time using PSS is longer than the ones using 2-mers. On the contrary, the number of reference sequences using PSS is less than the ones using 3-mers, so the computational time using PSS is shorter than the ones using 3-mers.

TABLE VII. ACCURACY COMPARISON OF MODELS USING PSS AND SPECTRUM KERNELS IN *D*1

	Training data	Testing data	Number of reference sequences	Computational time
PSS	71.667%	70%	250	0.1157
2-mers	65.833%	60%	144	0.0623
3-mers	70.833%	56.667%	1728	0.5558

Table VIII shows the accuracy comparison of models using PSS and spectrum kernels for *D*2. The table clearly exhibits the accuracy of models using PSS is better than the ones using other kernels. Although the accuracy of the model using PSS is slightly higher than the one using 3-mers, the computational time for the model using PSS is significantly less than using the one using 3-mers. It represents the proposed PSS kernel is very efficient to achieve the highest accuracy for training and testing data in *D*2.

TABLE VIII. ACCURACY COMPARISON OF MODELS USING PSS AND SPECTRUM KERNEL IN *D*2

	Training data	Testing data	Number of reference sequences	Computational time
PSS	67.317%	66.467%	18	98.3888
2-mers	65.158%	64.767%	144	249.6072
3-mers	66.608%	66.383%	1728	1677.1940

V. CONCLUSIONS

The sequence databases are commonly seen in our daily life, such as protein sequences, biological sequences, transaction sequences and so on. SVMs have been are widely applied for sequence classification due to they provide better

accuracy and more elastic performance. In SVMs, kernel method is one of the most important key to achieve high classification accuracy. Previous researchers proposed many kernel methods to sequence classification. Among them, the kernel using *k*-mers such as spectrum kernel and mismatch kernel is popular to resolve the sequence classification since *k*-mers provide a simple way to construct the reference sequence. However, if the number of symbols is too large, many worthless features in the kernel using *k*-mers will be involved

Instead of using *k*-mers, this paper proposed a pairwise sequence similarity (PSS) kernel that takes three sequential patterns which are *FSP*, *FCSP* and *FMSP* as reference sequences. Each pairwise sequence similarity score vector is computed by edit distance algorithm between sequences and reference sequence vector before building the SVM classifiers. Therefore, the major contributions of this research includes the proposal of a new PSS kernel using sequential patterns as reference sequences instead of the traditional *k*-mers. Moreover, the SVM classification model using PSS kernel can get the higher accuracy and more efficient classification.

The following suggestion is made to improve this method further. Currently, PSS score vectors are computed by EDA. Further study can use different sequence alignment approach algorithms, such as Needleman-Wunsch algorithm, Smith–Waterman algorithm and so on for evaluating the similarity between sequences and patterns. In addition, the cost parameter in EDA can be modified. In this research, the cost for deletion, exchange, and insertion are all set as 1. Those cost might affect the comparison result. In order to increase the classification accuracy, each sequence can be added a weight to represent the importance of each patterns. The higher weight reveals the pattern is more valuable. Finally, the proposed method can be applied not only to protein sequences or biological sequences but also to transaction sequences, travel route sequences, handwritten analysis and so on.

REFERENCES

[1] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," Proceedings of the 5th Annual Workshop on Computational Learning Theory, pp. 144-152, 1992.

[2] V. N. Vapnik, Statistical Learning Theory, 1998.

[3] B. Vanschoenwinkel and B. Manderick, "Appropriate kernel functions for support vector machine learning with sequences of symbolic data," Lecture Notes in Computer Science, vol. 3635, pp. 256-280, 2005.

[4] T. Howley and M. G. Madden, "The genetic kernel support vector machine: description and evaluation," Artificial Intelligence Review, vol. 24, no. 3-4, pp. 379-395, 2005.

[5] T. Jaakkola, M. Diekhans, and D. Haussler, "A discriminative framework for de¬tecting remote protein homologies," Journal of Computational Biology, vol. 7, no. 1-2, pp. 95-114, 2000.

[6] C. S. Leslie, E. Eskin, and W. S. Noble, "The spectrum kernel: a string kernel for SVM protein classification," Proceedings of Pacific Symposium on Biocomputing, pp. 566-575, 2002.

[7] C. S. Leslie, E. Eskin, A. Cohen, J. Weston, and W. S. Noble, "Mismatch string kernels for discriminative protein classification," Bioinformatics, vol. 20, no. 4, pp. 467-476, 2004.

[8] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins, "Text classification using string kernels," The Journal of Machine Learning Research, vol. 2, pp. 419-444, 2002.

[9] W. M. Campbell, J. R. Campbell, D. A. Reynolds, D. A. Jones, and T. R. Leek, "High-level speaker verification with support vector machines," Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 1, pp. 1-73, 2004.

[10] W. M. Campbell, J. P. Campbell, D. A. Reynolds, D. A. Jones, and T. R. Leek, "Phonetic speaker recognition with support vector machines," Advances in Neural Information Processing Systems, vol. 16, pp. 1377-1384, 2004.

[11] W. M. Campbell, J. P. Campbell, T. P. Gleason, D. A. Reynolds, and W. Shen, "Speaker verification using support vector machines and high-level features," IEEE Transactions on Audio, Speech, and Language Processing, vol. 15, no. 7, pp. 2085-2094, 2007.

[12] R. Agrawal and R. Srikant, "Mining sequential patterns," Proceedings of the 11th International Conference on Data Mining, pp. 3-14, 1995.

[13] J. Han, J. Pei, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M. C. Hsu, "Prefixspan: mining sequential patterns efficiently by prefix-projected pattern growth," Proceedings of the 17th International Conference on Data Engineering, pp. 215-224, 2001.

[14] X. Yan, J. Han, and R. Afshar, "CloSpan: mining closed sequential patterns in large datasets," Proceedings of the SIAM International Conference on Data Mining, pp. 166-177, 2003.

[15] P. Fournier-Viger, C. W. Wu, A. Gomariz, and V. S. Tseng, "VMSP: efficient vertical mining of maximal sequential patterns," Advances in Artificial Intelligence, pp. 83-94, 2014.

[16] C. W. Hsu and C. J. Lin, "A comparison of methods for multi-class support vector machines," IEEE transactions on Neural Networks, vol. 13, no. 2, pp. 415-425, 2002.

[17] R. K. Eichelberger and V. S. Sheng, "Does one-against-all or one-against-one improve the performance of multiclass classifications?" 27th AAAI Conference on Artificial Intelligence, pp. 1609-1610, 2013.

[18] E. L. Allwein, R. E. Schapire, and Y. Singer, "Reducing multiclass to binary: a unifying approach for margin classifiers," Journal of Machine Learning Research, vol. 1, pp. 113-141, 2001.

[19] J. Milgram, M. Cherier, and R. Sabourin, "One-against-one or one-against-all: which one is better for handwriting recognition with SVMs?," Proceedings of the 10th International Workshop on Frontiers in Handwriting Recognition, 2006.

Common Sense Knowledge, Ontology and Text Mining for Implicit Requirements

Onyeka Emebo[1,2], Aparna S. Varde[1], Olawande Daramola [2]
1. Department of Computer Science, Montclair State University, Montclair, NJ, USA.
2. Department of Computer and Information Sciences, Covenant University, Ota, Nigeria
emeboo@montclair.edu, vardea@montclair.edu, olawande.daramola@covenantuniversity.edu.ng

***Abstract*— The ability of a system to meet its requirements is a strong determinant of success. Thus effective requirements specification is crucial. Explicit Requirements are well-defined needs for a system to execute. IMplicit Requirements (IMRs) are assumed needs that a system is expected to fulfill though not elicited during requirements gathering. Studies have shown that a major factor in the failure of software systems is the presence of unhandled IMRs. Since relevance of IMRs is important for efficient system functionality, there are methods developed to aid the identification and management of IMRs. In this paper, we emphasize that Common Sense Knowledge, in the field of Knowledge Representation in AI, would be useful to automatically identify and manage IMRs. This paper is aimed at identifying the sources of IMRs and also proposing an automated support tool for managing IMRs within an organizational context. Since this is found to be a present gap in practice, our work makes a contribution here. We propose a novel approach for identifying and managing IMRs based on combining three core technologies: common sense knowledge, text mining and ontology. We claim that discovery and handling of unknown and non-elicited requirements would reduce risks and costs in software development.**

Keywords- Implicit Requirements, Common Sense Knowledge, Ontology, Text Mining, Requirement Engineering

I. INTRODUCTION

The challenge of identifying and managing implicit requirements has developed to be a crucial subject in the field of requirements engineering. In [7] it was stated "When critical knowledge, goals, expectations or assumptions of key stakeholders remain hidden or unshared then poor requirements and poor systems are a likely, and costly, consequence." With the relevance of implicit requirements (IMRs) being identified and related to the efficient functionality of any developed system, there have been different proposals as well as practical methodologies developed to aid the identification and management of IMRs. Common Sense Knowledge (CSK) is an area that involves making a computer or another machine understand basic facts as intuitively as a human would. It is an area in the realm of Knowledge Representation (KR) which involves paradigms for adequate depiction of knowledge in Artificial Intelligence (AI). The area of CSK is being researched for its use in identification and capturing of implicit requirements.

Since AI is aimed at enabling machines solve problems like humans, there is a need for common sense knowledge in AI systems to enhance problem-solving. This not only involves storing what most people know but also the application of that knowledge [8]. In software engineering, the development of systems must meet the needs of the intended user. However the very fact that CSK is common, not all knowledge and requirements that entail common sense, will be captured or expressed by the expected user. As Polanyi describes "We know more than we can tell". It is therefore the responsibility of the software developer to capture as well as manage the unexpressed requirements in the development of a suitable and satisfactory system. The application of Common Sense Knowledge can improve the identification as well as management of IMRs. Common Sense Knowledge CSK) is defined in [3] as a collection of simple facts about people and everyday life, such as "Things fall down, not up", and "People eat breakfast in the morning". In [7], the authors describe CSK as a tremendous amount and variety of knowledge of default assumptions about the world, which is shared by (possibly a group of) people and seems so fundamental and obvious that it usually does not explicitly appear in people's communications. CSK is mainly characterized by its implicitness.

From the literature, it is observed that a number of reasons have caused the emergence of implicit requirements some of which include; i) When a software organization develops a product in a new domain and ii) as a result of knowledge gap between developers and stakeholders due to the existence of implicit knowledge.

Given this background, we claim that CSK will aid in the identification of IMRs useful for domain-specific knowledge bases. This will be useful for storing domain concepts, relations and instances for onward use in domain related processing, knowledge reuse and discovery. Thus we build an automated IMR support tool based on our proposed framework for managing IMRs using common sense knowledge, ontology and text mining.

The rest of this paper is organized as follows: Section II presents core technologies. Section III reviews related work on IMRs. Section IV describes our automated IMR process framework. Section V describes the use and evaluation of the IMR support tool. Section VI gives the conclusions.

II. BACKGROUND

In this section, an overview of the concepts relevant in CSK, ontology, text mining and natural language processing is presented. This is useful in order to understand our proposed IMR framework later.

A. Common Sense Knowledge

Common Sense Knowledge (CSK) according to [3] is a tremendous amount and variety of knowledge of default assumptions about the world, which is shared by people and seems so obvious that it usually does not explicitly appear in

communications. Some characteristics of CSK as identified in the literature are as follows:

- *Share*: A group of people possess and share CSK.
- *Fundamentality*: People have a good understanding of CSK that they tend to take CSK for granted.
- *Implicitness*: People more often don't mention or document CSK explicitly since others also know it.
- *Large-Scale*: CSK is highly diversified and in large quantity.
- *Open-Domain*: CSK is broad in nature covering all aspects of our daily life rather than a specific domain.
- *Default*: CSK are default assumptions about typical cases in everyday life, so most of them might not always be correct.

Previous work on common sense knowledge includes the seminal projects Cyc [9] and WordNet [5], ConceptNet [20], Webchild [31] and the work by [14] and [24]. Cyc has compiled complex assertions such as every human has exactly one father and exactly one mother. WordNet has manually organized nouns and adjectives into lexical classes, with careful distinction between words and word senses. ConceptNet is probably the largest repository of common sense assertions about the world, covering relations such as hasProperty, usedFor, madeOf, motivatedByGoal, etc. Tandon et al. [14] automatically compiled millions of triples of the form <noun relation adjective> by mining n-gram corpora. Lebani & Pianta [24] proposed encoding additional lexical relations for commonsense knowledge into WordNet. WebChild contains triples that connect nouns with adjectives via fine-grained relations like hasShape, hasTaste, evokesEmotion, etc.

B. Ontology

The term ontology has different meanings. Ontology made and entrance in the field of computer science in the 1990s in association with Knowledge Acquisition. Different definitions have been given to the term "ontology". A basic definition of ontology was given in [37] as an explicit specification of a conceptualization". Author [39] explains it as a special kind of information object or computational artifact while [38] defined an ontology as a "formal specification of a shared conceptualization. Both definitions were merged by [12] hence defining an ontology as a "formal explicit specification of shared conceptualization". Ontologies provide a formal representation of knowledge and the relationships between concepts of a domain. They are used in the requirements specification to guide formal and unambiguous specification of the requirements, particularly in expressing concepts, relations and business rules of domain model with varying degrees of formalization and precision [26].

Formally an Ontology structure $\boldsymbol{O}$ can be defined as [18]

$$\boldsymbol{O} = \{\boldsymbol{C}, \boldsymbol{R}, \boldsymbol{A^o}\}$$

Where:

1. **C** is a set whose elements are called ***concepts***.
2. $\mathbf{R} \subseteq \boldsymbol{C\ X\ C}$ is a set whose elements are called ***relations***. For $\mathbf{r} = (c_1, c_2) \in \mathbf{R}$, it is written $\mathbf{r}(c_1) = c_2$.
3. $\mathbf{A^o}$ is a set of axioms on $\boldsymbol{O}$.

In recent times, there is an increased use of ontologies in software engineering. The use of ontologies has been proposed by different researchers' in. the field of requirements engineering and management According to [40] the increased use can be attributed to the following: (i) they facilitate the semantic interoperability and (ii) they facilitate machine reasoning. Researchers have so far proposed many different synergies between software engineering and ontologies. In Requirements Engineering (RE), ontology can be used for: 1) describing requirements specification documents and 2) to formally represent requirements knowledge [10]. Ontology is an important resources of domain knowledge, especially in a specific application domain. In the management of IMRs, ontology can provides shared knowledge which can be useful in the management of IMRs in similar or cross domain knowledge management. By conceptualizing domain knowledge including the identified implicit requirement, it enables the easy adoption and identification and also management of IMRs. This reduces enormous costs in requirement development process.in making "explicit specification" it aids the reduction of ambiguous requirements and incomplete definitions during the elicitation process [40]. By using such ontology, several kinds of semantic processing can be achieved in requirements analysis [31]. In this work, ontology is considered a valid solution approach, because it has the potential to facilitate formalized semantic description of relevant domain knowledge for identification and management of IMR.

C. Text Mining and Natural Language Processing

Text mining is the process of analyzing text to extract information that is useful for particular purposes [32]. [41] further expanded on the definition, stating that Text mining is the discovery and extraction of interesting, non-trivial knowledge from free or unstructured text everything from information retrieval (document or web site retrieval) to text classification and clustering, to entity, relation, and event extraction. It extracts information through the identification and exploration of interesting patterns [17]. Text mining has strong connections with Knowledge Management, data mining and Natural Language Processing (NLP). Authors in [41] describes NLP as an attempt to extract a fuller meaning representation from free text. In simple terms, it is figuring out who did what to whom, when, where, how and why. It 'typically makes use of linguistic concepts such as part-of-speech (noun, verb, adjective, etc.) and grammatical structure (either represented as phrases like noun phrase or prepositional phrase, or dependency relations like subject-of or object-of). NLP covers different disciplines from Linguistics to Computer Science and it's often closely linked with Artificial Intelligence. There are different definitions of NLP and they have evolved over the years. Natural Language Processing (NLP) generally refers to a range of theoretically

motivated computational techniques for analyzing and representing naturally occurring texts [7].

According to [4] it is made up of the following sub areas which are linked to linguistics; i) Morphology ii) Syntax iii)Semantics iv)Pragmatics

The core purpose of NLP techniques is to realize human-like language processing for a range of tasks or applications [8]. The core NLP models used in this research are part-of-speech (POS) tagging and sentence parsers [7]. POS tagging involves marking up the words in a text as corresponding to a particular part of speech, based on both its definition, as well as its context. In addition, sentence parsers transform text into a data structure (called a parse tree), which provides insight into the grammatical structure and implied hierarchy of the input text [7].

NLP is used for our purpose in analysis of requirements statements to gain an understanding of similarities that exist between requirements and/or identify a potential basis for analogy. NLP in combination with ontology enables the extraction of useful knowledge from natural language requirements documents for the early identification and management of potential IMRs.

III. Related work

Different researchers have proposed various ways for identification of IMRs. While some have presented applications, others have presented theoretical and conceptual frameworks and others take on the investigative approach in order to get real life views of practicing software engineers, requirements engineers and other specialists in the field on the practicality of stated theories, ideologies and concepts. Authors in [15] carried out a two part research aimed at identifying the impact of tacit and explicit knowledge transferred during software development projects. The first part involved an inductive, exploratory, qualitative methodology aimed at validating the tacit knowledge spectrum in software development projects. This involved unstructured interviews for data collection, and therefore assessment in a narrative form. The second part of this research involved the development of a conceptual framework of a model that supports future software development projects in their tacit to explicit knowledge transfers. [23] developed an approach based on a novel combination of grounded theory and incident fault trees. It focuses on Security Requirements. As a result of the threat landscape, there are new tacit knowledge which arise. This research proposes an approach to discover such unknown-knowns through multi-incident analysis. For this research an analysis and investigation method was used. It involved refining theoretical security models so that they remain pertinent in the face of a continuously changing threat landscape.

In a study carried out by [30], using a case study, the Knowledge Management on a Strategy for Requirements Engineering (KMoS-RE) which was designed to face the problem of management of tacit knowledge (in elicitation and discovery stage) and obtain a set of requirements that fulfill the clients' needs and expectations, was compared to requirements elicitation process proposed by MoProSoft; a Mexican software process model oriented to the specific needs of the software industry in Mexico. The results of this analysis showed that KMoS-RE seems to be more suitable than process proposed by MoProSoft. The KMoS-RE strategy improved the negotiation process and understanding about the domain and the software functionality requested and, the number of concepts and relationships was greater. KMos-RE strategy reduces the symmetry of ignorance between clients and users, and developers which facilitates the transference and transformation of knowledge and reduces increases the presence of unambiguous functional requirements.

Using requirements reuse for discovery and management of IMRs has been covered by a few studies. A study that draws on an analogy-making approach in managing IMRs is presented in [21]. This study proposes a system that uses semantic case-based reasoning for managing IMR. The model of a tool that facilitates the management of IMRs through analogy-based requirements reuse of previously known IMRs is presented. The system comprises two major components: semantic matching for requirements similarity and analogy-based reasoning for fine-grained cross domain reuse. This approach ensures the discovery, structured documentation, proper prioritization, and evolution of IMR, which is expected to improve the overall success of software development processes. However, as of now, this has not been adopted in a practical form. The work in [25] presents a model for computing similarities between requirements specifications to promote their analogical reuse. Hence, requirement reuse is based on the detection on analogies in specifications. This model is based on the assumption of semantic modeling abstractions including classification generalization and attribution. The semantics of these abstractions enable the employment of general criteria for detecting analogies between specifications without relying on other special knowledge. Different specification models are supported simultaneously. The similarity model which is relatively tolerant to incompleteness of specifications improves as the semantic content is enriched and copes well with large scale problems. Although the identification of analogies in requirements is essential, this study does not discuss the subject for the management of requirements. A method to highlight requirements potentially based on implicit or implicit-like knowledge is proposed in [2]. The identification is made possible by examining the origin of each requirement, effectively showing the source material that contributes to it. It is demonstrated that a semantic-level comparison enabling technique is appropriate for this purpose. The work helps to identify the source of explicit requirements based on tacit-like knowledge but it does not specifically categorize tacit requirement and its management. Also, in MaTREx [22], a brief review and interpretation of the literature on implicit knowledge useful for requirement engineering is presented. The authors describe a number of techniques that offer analysts the means to reason the effect of implicit knowledge and improve quality of requirements and their management. The focus of their work is on evolving tools and techniques to improve the management of requirements information through automatic trace recovery; discovering presence of tacit knowledge from tracking of presuppositions, non-provenance requirements etc. However, MaTREx still deals more with handling implicit knowledge.

Previous work on commonsense knowledge includes the Cyc project [9] with a goal to codify millions of pieces of common sense knowledge in machine readable form that enable a machine to perform human-like reasoning on such knowledge. Another source is WordNet [5] in which nouns and adjectives are manually organized into lexical classes, furthermore a distinction is made between words and word senses; yet its limitation is that there are no semantic relationships between the nouns and adjectives with the exception of extremely sparse attribute relations. Another prominent collection of commonsense is ConceptNet [20], which consists mainly of crowd sourced information. ConceptNet is the outcome of Open Mind Common Sense (OMCS) [6]. OMCS has distributed this CSK gathering task across general public on the Web. Through the OMCS website, volunteer contributors can enter CSK in natural language or even evaluate CSK entered by others.

Given this overview of the related literature, our proposed research stands out due to the fact that it introduces CSK for early identification and management of IMR. Moreover, it also embeds text mining and ontology to develop a support tool for managing IMRs. This is described next.

IV. The COTIR Framework

The architectural framework is made up of three core technologies: text mining/NLP, CSK and ontology as presented in Fig. 1. The core system functionalities are depicted as rectangular boxes, while the logic, data and knowledge artefacts that enable core system functionalities are represented using oval boxes. The detailed description of COTIR is given in below.

A. IMR Identification and Extraction

The part of the COTIR architecture that deals with knowledge representation and extraction is described in this section.

1) Data Preprocessing

A Software Requirements Specification (SRS) document that has been preprocessed serves as input to the framework. Preprocessing is a manual procedure that which entails extraction of boundary sentences from the requirements document and further representing images, figures, tables etc. in its equivalent textual format for use by the system.

2) NL Processor

The NL processor component facilitates the processing of natural language requirements for the process that enables feature extractor. The core natural language processing operations implemented in the architecture are i) Sentence selection, ii) Tokenization iii) Parts of speech (POS) tagging iv) Entity detection v) Parsing.

The Apache OpenNLP library [27] for natural language processing was used to implement all NLP operations.

3) Ontology Library

The ontology library and CSKB both make up the knowledge model of our framework. The ontology library serves as a storehouse for the various domain ontologies (.owl/.rdf). The domain ontologies are those that have been developed for specific purpose or those of business rules. The ontology library is implemented using Java Protégé 4.1 ontology API.

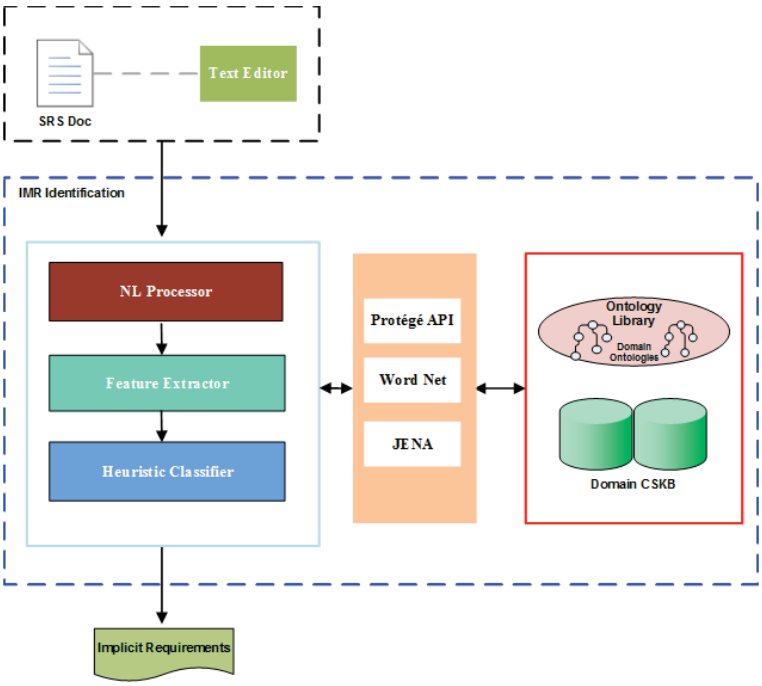


Fig. 1. Proposed COTIR Framework

4) Common Sense Knowledge Base(CSKB)

The common sense knowledge bases of WebChild and domain-specific KBs are used for enhanced identification of IMR for specific domain.

5) Feature Extractor

The feature extractor heuristic gives the underlying assumptions for identifying potential sources of IMR in a requirement document. Due to semantic features on which natural language text exist and by taking into account previous work done [11, 13, 16, 19, 28], the following characteristic features underline the significant aspects in a piece of text in terms of surface understanding that could possibly make a requirement implicit:

- Ambiguity such as structural and lexical ambiguity.
- Presence of vague words and phrases such as "to a great extent".
- Vague imprecise verbs such as "supported", "handled", "processed", or "rejected"
- Presence of weak phrases such as "normally", "generally".
- Incomplete knowledge.

V. IMR Support ToolUse and Evaluation

The COTIR framework illustrated in Fig. 1 is used to develop a support for the management of implicit requirements based on text mining, ontology and common sense knowledge. We now describe the use of this support tool for managing IMRs,

followed by a snap shot of the tool in Fig. 2 and 3 then its performance evaluation.

A. Use of the COTIR Tool

The process of using the COTIR tool developed in this work is as follows.

Step 1: Preprocess the source documents to get the requirements into text file format and devoid of graphics, images and tables.

Step 2: Select existing CSKB to be used for the identification of IMR.

Step 3: Import the requirement documents and domain ontology into COTIR environment.

Step 4: Click on the "analyze" function of the tool to allow the feature extractor identify potential sources of IMR in the requirement document.

Step 5: See the potential IMRs that are identified as well as their recommended possible explicit requirements.

Step 6: Seek the expert opinion on the IMRs; the experts could approve or disapprove the recommendations by adding/removing the recommendations through COTIR.

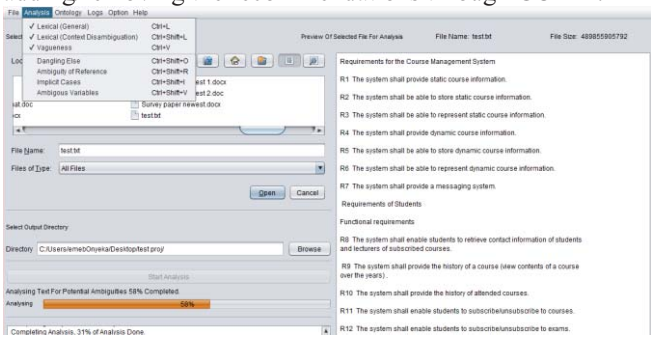
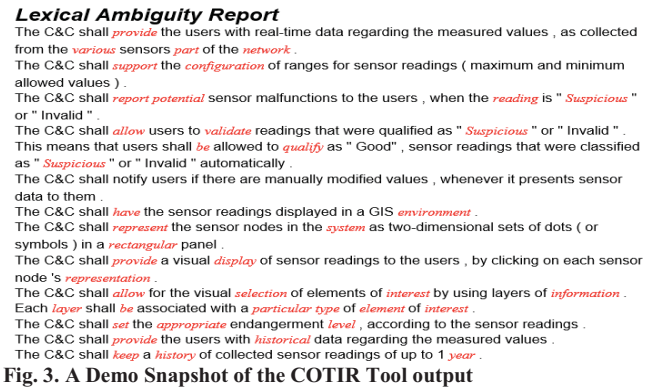


Fig. 2. A Demo Snapshot of the COTIR Tool input/Analysis screen



Lexical Ambiguity Report

The C&C shall *provide* the users with real-time data regarding the measured values , as collected from the *various* sensors *part* of the *network* .
The C&C shall *support* the *configuration* of ranges for sensor readings (maximum and minimum allowed values) .
The C&C shall *report potential* sensor malfunctions to the users , when the *reading* is " *Suspicious* " or " Invalid " .
The C&C shall *allow* users to *validate* readings that were qualified as " *Suspicious* " or " Invalid " .
This means that users shall *be* allowed to *qualify* as " Good" , sensor readings that were classified as " *Suspicious* " or " Invalid " automatically .
The C&C shall notify users if there are manually modified values , whenever it presents sensor data to them .
The C&C shall *have* the sensor readings displayed in a GIS *environment* .
The C&C shall *represent* the sensor nodes in the *system* as two-dimensional sets of dots (or symbols) in a *rectangular* panel .
The C&C shall *provide* a visual *display* of sensor readings to the users , by clicking on each sensor node 's *representation* .
The C&C shall *allow* for the visual *selection* of elements of *interest* by using layers of *information* .
Each *layer* shall *be* associated with a *particular type* of *element* of *interest* .
The C&C shall *set* the *appropriate* endangerment *level* , according to the sensor readings .
The C&C shall *provide* the users with *historical* data regarding the measured values .
The C&C shall *keep* a *history* of collected sensor readings of up to 1 *year* .

Fig. 3. A Demo Snapshot of the COTIR Tool output

For evaluation of the COTIR (Commonsense Ontology and Text-mining of Implicit Requirement) tool developed, we conduct an assessment of its performance using requirements specification. The objectives of the evaluation are as follows: (1) to assess the performance of the tool by human experts, (2) to determine its usefulness as a support tool for implicit requirements management within an organization. (3) to identify areas of possible improvement in the tool.

B. Performance Evaluation Procedure

The evaluation makes use of the following sets of requirements specification: i) Course Management System (CMS) [33], this project was developed for use at the University of Twente course management system. The requirements for CMS describes basic functionality like course enrollment, course material and roster upload, student grading and e-mails communication. ii) EMbedded Monitoring Project [34], the EMMON project is a European Research and Development (R&D) project. The project captures requirements for smart locations and ambient intelligent environments (smart cities, smart homes, smart public spaces, smart forests, etc.) iii) Tactical Control System (TCS) requirements [35], This project was designed for the Naval Surface Warfare Center-Dahlgren Division and Joint Technology Center/System Integration Laboratory, Research Development and Engineering Center, U.S.

This three requirements specification documents were code named R1, R2, R3 as shown in the evaluation table II.

We use the following metrics to assess the performance of the system. Precision (P), Recall (R), F-measure (F).

$$R = \frac{TP}{(TP+FN)} \qquad P = \frac{TP}{TP+FP}$$

$$F = \frac{PR}{P+R}$$

In these formulas, TP, TN, FP and FN are as follows.

TP (true positives): number of requirements judged by both the expert and tool as being implicit

TN (true negative): number of requirements judged by both the expert and tool as not being implicit

FP (false positive): number of requirements judged by the tool as implicit and by the expert as not implicit

FN (false negatives): number of requirements judged by the tool as not implicit and by the expert as implicit.

A group of subjects were asked to mark implicitness in the requirement document and also use the tool.

The Subjects are a group of computing professionals, comprising software developers, academics and research students. They were given the following instructions: 1) for each specified requirement, mark each requirement based on its implicit nature (noting that a requirement may contain more than one form of implicitness). 2) For each requirement specify the degree of criticality of each implicitness on a scale of 1 to 5. (1 being least critical to 5 being most critical).

Table I shows a sample identification form. The type of implicitness includes i) Ambiguity (A) ii) Incomplete Knowledge (IK) iii) Vagueness (V) iv) Others

C. Results of Performance Evaluation

Table II shows the recall, precision and F-scores computed for the tool relative to eight experts' (E1–E8) evaluation.
For a detection tool, the recall value is definitely more important than precision. In the ideal case, the recall should be 100%, as it would allow to relieve human analysts from the clerical part of document analysis [36]. For our tool with an average recall value of about 73.7%, it show that the tool is fit for practical use, as it marks six out of eight IMR detected by humans and is consistent with best practices. The average precision is 68.22% which shows the percentage of IMR judged by experts that was also retrieved by the tool is good and still consistent with best practices. The average F-score which is a harmonic mean of Precision and Recall is 70.3%. Which shows that the result of the tool's performance was good. As for the IMRs marked by human evaluators but missed by the tool, manual examination has shown that they represent implicit factors where we could not identify explicit patterns that would allow to automate IMR detection. Our observation from the simulation experiment (see Figs. 3.) is that the performance of the tool also depends significantly on the quality of the domain ontology and CSKB.

Table I: Sample Identification Form

S/N	Requirement	Type of Implicitness	Criticality
1	The C&C shall provide the users with real--time data regarding the measured values, as collected from the various sensors part of the network.	(A)	1 2 3 4 5
		(IK)	1 2 3 4 5
		(V)	1 2 3 4 5
		(O)	1 2 3 4 5
2	The C&C shall support the configuration of ranges for sensor readings (maximum and minimum allowed values).	(A)	1 2 3 4 5
		(IK)	1 2 3 4 5
		(V)	1 2 3 4 5
		(O)	1 2 3 4 5
3	The C&C shall report potential sensor malfunctions to the users, when the reading is "Suspicious" or "Invalid".	(A)	1 2 3 4 5
		(IK)	1 2 3 4 5
		(V)	1 2 3 4 5
		(O)	1 2 3 4 5

Table II: Recall, Precision and F-Score metrics from 8 experts (E1-E8)

	Requirements	E1	E2	E3	E4	E5	E6	E7	E8	Average
Precision	**R1**	75	75	75	69.23	66.67	83.33	50	91.67	73.24
	R2	66.67	58.33	33.33	58.33	83.33	58.33	75	75	63.54
	R3	68.75	81.25	56.25	75	43.75	86.67	50	81.25	67.87
Average										**68.22**
Recall	**R1**	90	90	75	90	80	83.33	75	78.57	82.74
	R2	66.67	70	66.67	58.33	76.92	70	69.23	75	69.1
	R3	73.33	72.22	64.29	66.67	58.33	81.25	61.54	76.47	69.26
Average										**73.7**
F-Score	**R1**	81.82	81.82	75	78.26	72.73	83.33	60	84.62	77.2
	R2	66.67	63.63	44.44	58.33	80	63.63	72	75	65.46
	R3	70.97	76.47	60	70.59	50	83.87	55.17	78.79	68.23
Average										**70.3**

VI. Conclusions

In conclusion, the ability to automatically identify and manage IMRs will mitigate many risks that can adversely affect system architecture design and project cost. This research work evolves a systematic tool support framework which uses common sense knowledge that can be integrated into an organizational Requirement Engineering procedure for identifying and managing IMR in systems development process. This is a direct response to problems in the practice of many organizations that have not been addressed by existing requirements management tools. Hence, this work addresses the problem of identifying IMRs in Requirements documents and its further management. The novelty of this work is that integrates three core technologies, namely, common sense knowledge, ontology and text mining to propose an automated IMR framework. Another significant contribution is that a support tool is developed based on the proposed framework and is helpful in domain-specific contexts. This work would useful to AI scientists and software engineers. Its targeted applications include providing software requirements for various AI systems, where common sense is useful in automation.

Acknowledgments

This work was conducted when Onyeka Emebo was a Fullbright Scholar at Montclair State University, USA, visiting from Covenant University, Nigeria. The authors thank the source of the Scholarship for this funding.

REFERENCES

[1] A. Parameswaran,: Capturing Implicit Requirements (02-08-2011), http://alturl.com/emeej

[2] A. Stone, and P. Sawyer, : Identifying tacit knowledge-based requirements. In Software, IEE Proceedings-, vol. 153, no. 6, pp. 211-218. IET, 2006.

[3] Zang, Liang-Jun, Cong Cao, Ya-Nan Cao, Yu-Ming Wu, and C. A. O. Cun-Gen. "A survey of commonsense knowledge acquisition." Journal of Computer Science and Technology 28, no. 4 (2013): 689-719.

[4] Copestake, Natural Language Processing: Part 1 of Lecture Notes. Cambridge: Ann Copestake Lecture Note Series, 2003.44

[5] C.D. Fellbaum, G.A. Miller (Eds.): WordNet: An Electronic Lexical Database. MIT Press, 1998.

[6] Singh P, Lin T, Mueller E, Lim G, Perkins T, Zhu W. Open mind common sense: Knowledge acquisition from the general public. In Proc. Conf. Cooperative Information Systems, Oct.30-Nov.1 2002, pp.1223-1237.

[7] Choi, F. Y. (2000, April). Advances in domain independent linear text segmentation. In Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference (pp. 26-33). Association for Computational Linguistics.

[8] Chowdhury, G. G. (2003). Natural language processing. Annual review of information science and technology, 37(1), 51-89.

[9] Douglas B. Lenat: CYC: A Large-Scale Investment in Knowledge Infrastructure. Comm. of the ACM 38(11), pp. 32-38, 1995.

[10] Decker, B., Ras, E., Rech, J., Klein, B., & Hoecht, C. (2005, November). Self-organized reuse of software engineering knowledge supported by semantic wikis. In Proceedings of the Workshop on Semantic Web Enabled Software Engineering (SWESE) (p. 76).

[11] Fabbrini, F., Fusani, M., Gnesi, S., & Lami, G. (2001, June). An automatic quality evaluation for natural language requirements. In Proceedings of the Seventh International Workshop on Requirements Engineering: Foundation for Software Quality REFSQ (Vol. 1, pp. 4-5).

[12] Gruninger, M., & Lee, J. (2002). Ontology-applications and design. Communications of the ACM, 45(2), 39-41.

[13] Kamsties, E., Berry, D. M., Paech, B., Kamsties, E., Berry, D. M., & Paech, B. (2001, July). Detecting ambiguities in requirements documents using inspections. In Proceedings of the first workshop on inspection in software engineering (WISE'01) (pp. 68-80).

[14] N. Tandon, G. de Melo, G. Weikum: Deriving a Web-Scale Common Sense Fact Database. AAAI 2011.

[15] Dreyer, H., Wynn, M. G., & Bown, G. R. (2015). Tacit and Explicit Knowledge in Software Development Projects: Towards a Conceptual Framework for Analysis. In eKnow 2015 7th International Conference on Information, Process and Knowledge Management (No. A, pp. 49-52). ThinkMind.

[16] Lami, G., Gnesi, S., Fabbrini, F., Fusani, M., & Trentanni, G. (2004). An automatic tool for the analysis of natural language requirements. Informe técnico, CNR Information Science and Technology Institute, Pisa, Italia, Setiembre.

[17] Feldman, R., & Sanger, J. (2007). The text mining handbook: advanced approaches in analyzing unstructured data. Cambridge University Press.

[18] Maedche, A. (2012). Ontology learning for the semantic web (Vol. 665). Springer Science & Business Media.

[19] Meyer, B. (1985). On formalism in specifications. IEEE software, 2(1), 6.

[20] R. Speer, C. Havasi: Representing General Relational Knowledge in ConceptNet 5. LREC 2012.

[21] O. Daramola, T. Moser, G. Sindre, and S. Biffl,: Managing Implicit Requirements Using Semantic Case-Based Reasoning Research Preview. REFSQ 2012, LNCS 7195, pp. 172–178, Springer-Verlag Berlin Heidelberg (2012).

[22] R. Gacitua, B. Nuseibeh, , P. Piwek, , A.N. de Roeck, , M. Rouncefield, , P. Sawyer, , A. Willis, and H. Yang, "Making Tacit Requirements Explicit", Second International Workshop on Managing Requirements Knowledge (MaRK'09) 2009.

[23] Rashid, A., Naqvi, S. A. A., Ramdhany, R., Edwards, M., Chitchyan, R., & Babar, M. A. (2016, May). Discovering unknown known security requirements. In Proceedings of the 38th International Conference on Software Engineering (pp. 866-876). ACM.

[24] G.E. Lebani, E. Pianta: Encoding Commonsense Lexical Knowledge into WordNet. Global WordNet Conference 2012.

[25] Spanoudakis, G. (1996). Analogical reuse of requirements specifications: A computational model. Applied Artificial Intelligence, 10(4), 281-305.

[26] Sugumaran, V., & Storey, V. C. (2002). Ontologies for conceptual modeling: their creation, use, and management. Data & knowledge engineering, 42(3), 251-271.

[27] Welcome to Apache OpenNLP (24/10/2012) http://opennlp.apache.org/

[28] Wilson, W. M., Rosenberg, L. H., & Hyatt, L. E. (1997, May). Automated analysis of requirement specifications. In Proceedings of the 19th international conference on Software engineering (pp. 161-171). ACM.

[29] Yatskevich, M., & Giunchiglia, F. (2004). Element level semantic matching using WordNet. In Meaning Coordination and Negotiation Workshop, ISWC

[30] Sánchez, K. O., Osollo, J. R., Martínez, L. F., & Rocha, V. M. (2015). Requirements engineering based on knowledge: a comparative case study of the KMoS-RE strategy and the DMS process. Revista Facultad de Ingeniería, (77), 88-94.

[31] Tandon, N., de Melo, G., Suchanek, F., & Weikum, G. (2014). Webchild: Harvesting and organizing commonsense knowledge from the web. In Proceedings of the 7th ACM international conference on Web search and data mining (pp. 523-532). ACM.

[32] Gharehchopogh, F. S., & Khalifelu, Z. A. (2011). Analysis and evaluation of unstructured data: text mining versus natural language processing. In Application of Information and Communication Technologies (AICT), 2011 5th International Conference on (pp. 1-4). IEEE.

[33] Abma, B. J. M. "Evaluation of requirements management tools with support for traceability-based change impact analysis." Master's thesis, University of Twente, Enschede (2009).

[34] Software Requirements Specification – EMMON (12-07-2010) http://www.artemis-emmon.eu/deliverables/FP7-JU-EMMON-2010-DL-WP7-003-D7.1-software-requirements-specification-document.pdf.

[35] TCS Software Requirements Specification (02-12-1999) https://fas.org/irp/program/collect/docs/sss_20.pdf.

[36] Kiyavitskaya, N., Zeni, N., Mich, L., Berry, D.M.: Requirements for tools for ambiguity identification and measurement in natural language requirements specifications. Requir. Eng. 13 (2008) 207–239

[37] Guber, T. (1993). A Translational Approach to Portable Ontologies. Knowledge Acquisition, 5(2), 199-229.

[38] Borst, W. N. (2002). Construction of engineering ontologies. 1997. Centre of Telematica and Information Technology, University of Tweenty: Enschede, The Netherlands.

[39] Guarino, N., Oberle, D., & Staab, S. (2009). What is an ontology?. In Handbook on ontologies (pp. 1-17). Springer Berlin Heidelberg.

[40] Castañeda, V., Ballejos, L., Caliusco, M. L., & Galli, M. R. (2010). The use of ontologies in requirements engineering. Global journal of researches in engineering, 10(6).

[41] Kao, A., & Poteet, S. R. (Eds.). (2007). Natural language processing and text mining. Springer Science & Business Media.

KDPMEL: A Knowledge Discovery Process Modeling and Enacting Language

Hesham A. Mansour
FJA-US, Inc., 1040 Avenue of the Americas, 4th Floor, New York, NY 10018, USA

Abstract - *Knowledge Discovery in Databases (KDD) is a complex process that is highly interactive and iterative. Managing the process and its underlying resources and interacting activities is a very complex task. Today, KDD projects are typically approached in an unstructured, ad hoc manner due to a lack of formal methods and methodologies for their development. The similarities between KDD processes and software development processes suggest that approaches used to manage the development of software processes, such as process programming and process-centered support environments, are also applicable and in fact advantageous to KDD processes. This paper proposes the Knowledge Discovery Process Modeling and Enacting Language (KDPMEL) that is designed specifically to provide both the syntax and semantics needed to capture and execute KDD processes. KDPMEL combines aspects of both process programming and KDD to represent KDD processes precisely as process programs. Along with KDPMEL, an Integrated Development Environment (IDE) for KDPMEL is proposed to enable and facilitate the development and execution of KDPMEL programs. We illustrate and evaluate KDPMEL by representing and executing illustrative examples of KDD activities and tasks.*

Keywords: Data Mining, Knowledge Discovery in Databases (KDD), KDD Process, Process Programming

1 Introduction

The complexities of KDD processes have been recognized and as a result, various approaches and methodologies have been proposed [1] to provide user guidance and support for the development of KDD processes.

Unfortunately, while many KDD support systems have been proposed and prototyped, the support they provide either targets individual activities performed within the process---*activity-oriented* support---or is based on architectures that support a predefined generic process model---*KDD support environments*. There have been a few proposals that apply *process-oriented* support to KDD. Among these, only [3] uses a process language approach based on Little-JIL to explicitly represent the coordination aspect of KDD processes. Each of these approaches has certain limitations and lacks either the proper formalism and/or the suitable semantics needed to explicitly represent and effectively support the performance of KDD processes.

In activity-oriented support, the process concept is neither enforced nor endorsed and is only represented in documentation and guidelines. In contrast, while the process concept is adopted by KDD support environments, it is limited to a generic form that is described by the major phases and tasks of a designated process model. In addition, these environments usually support particular KDD techniques and have a prescribed set of supporting tools that are typically built in the environment or tightly integrated with it. Although the process-oriented support presented in [3] is very promising, it concentrates on supporting only the coordination aspect of the process. In addition to the discovered deficiencies in Little-JIL, only the simplest processes can be modeled visually using Little-JIL. For additional information about the different approaches for supporting KDD processes and their limitations, see [1].

The similarities between KDD processes and software processes suggest that approaches used to manage and support the development of software processes are also applicable and in fact advantageous to KDD processes. Although some researchers [3], [6]-[8] have recognized these similarities, none (to our knowledge) has proposed a comprehensive approach for supporting KDD processes through a dedicated KDD process programming language that can combine aspects from both process technology and KDD to explicitly represent and support KDD processes more accurately. This language represents a core component of a much larger process-centered support environment that can enable and facilitate the modeling, management, and enactment of KDD process programs written in this language.

This paper proposes the Knowledge Discovery Process Modeling and Enacting Language (KDPMEL) that is designed specifically to provide both the syntax and semantics needed to capture and execute KDD processes. KDPMEL provides predefined language constructs for modeling the resources of a process, encoding its steps and specifying their sequencing, invoking external tools, and specifying the usage of resources by the steps of the process. KDPMEL combines aspects of both process programming and KDD to represent KDD processes precisely as process programs. KDPMEL supports modeling KDD tasks at different levels of detail and

abstraction in order to specify both generic and specialized tasks. KDPMEL provides special control constructs to explicitly model task dependencies on other tasks. This feature is particularly important for KDD and is intended to effectively manage the dependencies between KDD techniques. Having these dependencies explicitly defined can assure that they are appropriately handled. We believe that the above mentioned features precisely suit KDD and their existence in a process language that combines process aspects along with KDD aspects is novel for implementing KDD processes.

Our philosophy in designing a process language for KDD instead of utilizing or adapting an existing process language is that we want the language to be highly expressive for the essential aspects of KDD and not just the process-aspect. The paper is structured as follows. This section provides background information and the motivation for our work. Section 2 presents KDPMEL and illustrates its usage in modeling KDD processes. Section 3 outlines an Integrated Development Environment (IDE) for KDPMEL. Section 4 briefly evaluates KDPMEL and provides some lessons earned during its implementation. Section 5 concludes the paper.

2 The Knowledge Discovery Process Modeling and Enacting Language (KDPMEL)

KDPMEL is a novel process programming language for modeling and enacting KDD activities along with their resources, interactions, coordination, and dependencies. The process aspects of the language such as process structuring, task ordering, and artifact management are similar to those found in general process languages, such as Little-JIL [3] and PML [4]. The KDD aspects of the language are specific features for modeling KDD artifacts, tools, and tasks.

KDPMEL provides multiple granularity levels for process decomposition that is particularly suited to KDD. The first level allows for organizing the process into a number of phases (Lifecycle level). The second level allows for defining the generic tasks of each phase (Generic Task level). Finally, the third level allows for defining the specialized tasks for a generic task (Specialized Task level).

A process program written in KDPMEL is organized into three major sections that specify the resources (*artifacts*, *roles*/*actors*, and *tools*) of the process, general information about the process (*goal*, *input*, *outcome*, and *assessment*), and the steps (*activity*, *action*, and *command*) of the process along with their sequencing (*sequence*, *parallel*, *choice*, and *loop*) and dependencies (*disallow*, *require*, and *enable*).

2.1 Language Goals

The major goals of the KDPMEL are the *simplicity*, *flexibility*, *expressiveness*, and *generality*.

2.2 Language Approach

KDPMEL combines both the graphical and process language approaches. Modeling with KDPMEL employs a number of graphical modeling editors on top of a textual process programming language designed specifically for KDD. The goal of the modeling editors is to facilitate the construction and presentation of certain components of the process, as represented by four types of graphs that display the overall process (Process Graph), the resources of the process (Resources Graph), a graph for each activity (Activity Graph), and a graph for each action (Action Graph). Furthermore, a read-only graph that shows the progress of the artifacts within the process (Artifact Flow Graph) is provided. The Process Graph indicates process information such as goal, input, outcome, etc. The Resources Graph shows the artifacts, roles/actors, and tools of the process. Each Activity Graph shows the activity's constituent actions while each Action Graph provides information such as tools utilized by the action, the artifacts consumed and produced by the action, and the actor assigned to the action. In addition to the graphical editors, a number of form-based editors exist (Process, Activity, Artifact, Role/Actor, and Tool Forms) to present and update certain information that is best shown and updated in a form-based style.

Combining different types of editors and views in source-based, graph-based, and form-based styles is novel and allows both technical and non-technical users to participate in the modeling phase of the process. Fig. 1 illustrates the process modeling approach adopted by KDPMEL.

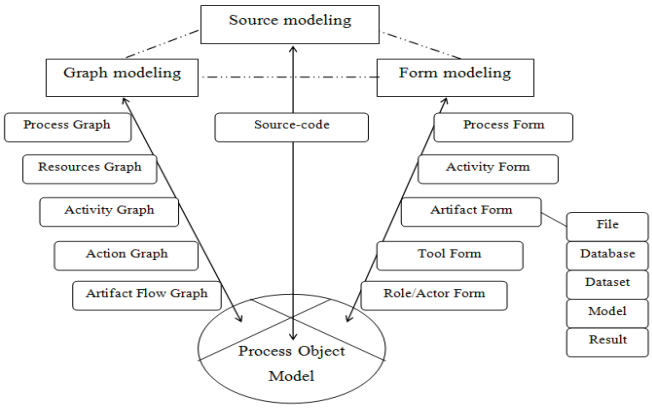


Fig. 1. KDPMEL Process Modeling Approach

As illustrated in Fig. 1, the process object model (the internal object-oriented representation of the process model) is described using source-based modeling, where all the process details can be specified using the KDPMEL source-code editor. The graph-based and form-based modeling approaches are used for certain process parts that are best presented (e.g., overall process structure and control flow) and updated (e.g., artifacts description) using these approaches. Alternatives exist. For instance, the process object model can be created and updated by either the source-based or graph-based approaches, whereas the form-based

approach can only update the process object model. The process object model is translated into its source-based, graph-based, and form-based representations through a number of model visitors, which are based on the object-oriented visitor design pattern [15], that navigate the model and perform the appropriate translation. The source-code model visitor performs complete translation for the model to its source-code representation. The graph model visitors create graphs for the process, resources, each activity, each action, and a read-only graph for the flow of artifacts in the process. The form model visitors create forms for the process and for each activity, action, artifact (File, Database, Dataset, Model, and Result), role/actor, and tool. In the figure above, the dashed lines between the different modeling approaches indicate the translations between these approaches.

We believe that this hybrid modeling approach combines the benefits of the underlying approaches and enables specification of high-level process models as well as more complex ones in a manner that is convenient for both technical and non-technical users.

A KDPMEL process program can be developed from scratch using either the source-based or graph-based modeling approach and can be updated using any approach. Any update to the process program in any representation is reflected to the other representations.

The adopted graphical modeling approach tries to avoid ambiguities and mistakes. It also tries to ease the graphical modeling development. Graphical components that are new are inserted into a graph by a drag and drop approach. Graphical components that are already defined but need to be referenced in a graph are inserted into the graph by selecting from a menu that displays appropriate graphical components for the current context. For example, to insert a new actor resource component into the Resources Graph Editor, this component is dragged from a palette and dropped into the editor as shown in Fig. 2.

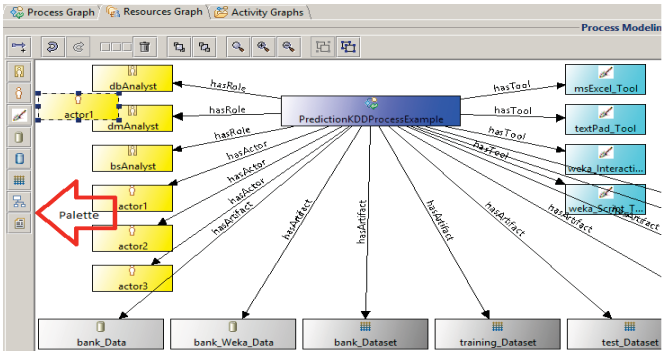


Fig. 2. Inserting an Actor Resource component in The Resources Graph Editor

An example for inserting an actor performer component into the Action Graph Editor is shown in Fig. 3, where a context menu is used to select the type of component to be inserted and another menu is used to select the value of that component.

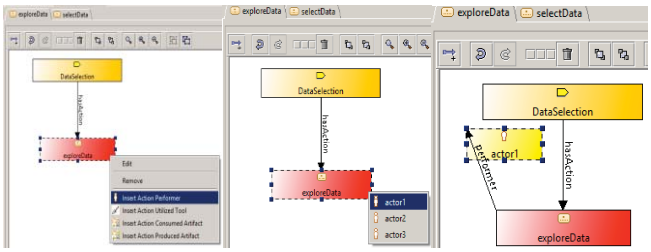


Fig. 3. Inserting an Actor Performer component in the Action Graph Editor

This approach avoids modeling mistakes such as inserting an actor performer component for an actor that was never defined in the resources, if a drag and drop approach were to be used. Even if that actor was already defined in the resources, mistakes can still happen such as specifying the wrong actor name, which was different than what was defined. Moreover, it avoids ambiguities in specifying, for instance, which artifact is consumed and which is produced by the action. If a drag and drop approach were to be used, this case would require the modeler to distinguish between the action's consumed artifact and the other produced one by specifying the correct association name for each artifact, which is more work and can lead to mistakes. Designating a separate association type for each artifact is overwhelming and complicates the graphical modeling.

2.3 KDPMEL Meta-Model (Abstract Syntax)

KDPMEL is defined in terms of a meta-model [11] based on the OMG's SPEM [16] and CWM [10] meta-models, which represents the abstract syntax and static semantics of the language. The abstract syntax is represented by the Unified Modeling Language (UML) [9] class diagrams showing the language metaclasses and their relationships.

2.4 KDPMEL Concrete Syntax

The concrete syntax of KDPMEL is provided in two flavors, textual and graphical, to serve different purposes. The textual concrete syntax (grammar) is useful when specific complex details must be specified. The graphical concrete syntax is easy to understand and use, and is useful for communicating structural and higher level views of the process models created by the language. Also, a form-based interface is provided for process components to allow for presenting and updating the properties of these components.

2.4.1 KDPMEL Textual Concrete Syntax (Grammar)

KDPMEL meta-model provides process specific entities such as *Process*, *Activity*, and *Action*. The *Process* entity represents the entire process and is intended to be used as a container for all the process components. The *Activity* entity represents a composite activity in the process that can be further decomposed into smaller units. The *Action* entity represents a primitive task in the process. The syntax for defining these constructs is given by the following rules:

<process> ::= "**process**" <IDENTIFIER> "{"…"}"

<activity> ::= "**activity**" <IDENTIFIER> "{"…"}"

<action> ::= "**action**" <IDENTIFIER> "{"…"}"

For additional information about KDPMEL syntax, see [1].

2.4.2 Modeling KDD Task Dependencies

A KDD task may have dependencies with other tasks in the process. It may *require*, *disallow*, or *enable* other tasks.

The experiment conducted by Jensen et al. [3] on the subject of coordinating interdependent KDD tasks using Little-JIL showed that even a state-of-the-art process language such as Little-JIL lacks the semantics to describe some simple cases easily. Little-JIL lacks semantics for establishing dependencies among tasks and consequently a Little-JIL program must be written in a manner that conforms to these dependencies, which are represented in the structure of the program. A major drawback for this approach is that any change in the process specifications would need substantial modifications for the process program. The authors have acknowledged that adding to Little-JIL process specifications is not as simple as it was hoped to be, at least in the domain of KDD, and requires substantial revisions in process specifications due in part to the interdependence of KDD techniques and their effects. They also stated that changes to the Little-JIL language may be essential to overcome this deficiency.

The KDPMEL approach for explicitly modeling task dependencies is not only more flexible in accommodating changes to the process specifications, but it is more expressive in representing process specifications. The following KDPMEL example specifies a *disallow* dependency between the Three Group Regression (TGR) technique and the Parametric Significance Test:

```
activity LinearRegression {
  choice buildRegressionModel {
    action constructLSRModel {…}
    action constructTGRModel {…
      dependency {
        require …; enable …;
        disallow parametricSignificanceTest;
      }
    }
  }
  choice testRegressionModel {
    action parametricSignificanceTest {…}
    action randomizationTest {…}
  }
}
```

2.4.3 Modeling Specialized KDD Tasks

KDPMEL represents specialized KDD tasks through external commands that are modeled in the program and can be validated and executed through a plug-in mechanism for the tools of these commands.

The following is an example of a KDPMEL action that uses a command to build a decision tree model with the C4.5 (J48) classifier algorithm [12] of the WEKA data mining framework [13]:

```
action buildDecisionTreeModel {
  consume sampleDataset;
  produce sampleDecisionTreeModel;
  performer dmAnalyst;
  utilize {
    call wekaTool {
      command build_C45_Weka_Classifier {
        kind Modeling;
        input sampleDataset;
        output sampleDecisionTreeModel;
        operation"weka.classifiers.trees.J48";
        parameters "-C 0.25 -M 2";
      }
    }
  }
}
```

2.4.4 Modeling Consistency Rules and Constraints for KDD Tasks

The ability to explicitly model and enforce requirements and acceptance criteria of KDD tasks is important for making the KDD process more understandable, evaluating its correctness, assuring its consistent execution, and validating its results.

Requirements and acceptance criteria can be defined for KDPMEL tasks using *pre-conditions* and *post-conditions* constructs to represent constraints that guard entry (task requirements) into and exit (task acceptance criteria) from a task. These constraints are defined by logical expressions over the process artifacts to check their attributes and states.

Artifact State Expression

The predicate *hasState(artifact, state)* checks if an artifact has a particular state. For example, the expression: *hasState(bank_Data, Prepared)* checks if the *bank_Data* artifact has the *Prepared* state.

Artifact Attribute Expressions

Two types of expressions are used to check the attributes of an artifact. The first checks if an attribute is defined in an artifact using the *isDefined(attributeList)* predicate. For example, the expression: !*isDefined(bank_Dataset.id, bank_Dataset.save_act)* checks if the *id* and *save_act* attributes are not defined in the *bank_Dataset* artifact. The second expression type is an attribute comparison expression that compares the value of an attribute with another or with a literal value. For example, the expression: *bank_training_Dataset.instancesPct* == 0.75 && *bank_test_Dataset.instancesPct* == 0.25 checks if the *bank_training_Dataset* artifact has 75% of the instances and the *bank_test_Dataset* artifact has 25% of the instances.

Consistency Rules

Consistency rules are represented as constraints on KDPMEL tasks in the form of pre-conditions and post-conditions that regulate the work of the tasks over the artifacts. In order to maintain valid state changes for an artifact, there should be a *hasState* predicate for that artifact in both the pre-conditions and post-conditions.

2.5 KDPMEL Semantics

Control Flow and Ordering

KDPMEL provides a variety of constructs for specifying the control flow among activities and actions. By using these control constructs, the process control flow can be described at different levels of detail. The activities within a process and the actions within an activity can be grouped using one of the control constructs *sequence*, *parallel*, *choice*, or *loop*. Additionally, activities may be decomposed into a hierarchy of sub-activities and actions.

Ordering

Both the activities within a process and the actions within an activity can be grouped using *sequence*, *parallel*, *choice*, or *loop*. The default ordering is *sequence*.

Sequence

A *sequence* control construct specifies that the execution of its group of tasks (i.e., activities or actions) is performed in the order that they are written in the process program.

Choice

A *choice* control construct specifies that exactly one task is performed from its group of tasks. The decision for determining which task to execute is mainly handled by checking the requirements of each task in the choice. If the requirements for only one of the tasks in the choice are satisfied, then this task is executed. If there are no requirements or if the requirements of two or more tasks are satisfied, then the actor performing the choice must select which task to execute.

Parallel

A *parallel* control construct specifies that its group of tasks can be performed simultaneously.

Loop

A *loop* control construct specifies that its group of tasks can be performed repeatedly. The decision to stop or continue the loop after completing the last task in the loop is handled by checking the requirements of the first task following the loop. The following task may have explicit preconditions and/or implicit requirements such as needed artifacts.

Hierarchical Task Decomposition

An *activity* construct can be decomposed into smaller units of sub-activities and/or actions. This decomposition creates a tree structure of activities and actions. The execution of these tasks is performed depth-first, where the execution of a node in the tree is performed by executing its children in left to right order.

KDD processes may have highly variable control requirements including both strict and loose control. Strict control, in the forms of hierarchical task decomposition and the sequence control construct (flat task decomposition), is needed to specify that certain tasks must be executed in a given order while loose control, for instance, using the *choice* control construct is needed to specify different execution alternatives among the tasks. Both types of decompositions (Flat and Hierarchical) are used to decompose high-level process activities (e.g., the phases of the CRISP-DM [2] process) into lower-level activities and primitive actions (e.g., the generic tasks of the CRISP-DM process), which is a natural way of organizing the process tasks using a divide-and-conquer approach.

Dependency Control Constructs

The dependency control constructs *disallow*, *require*, and *enable* can be associated with an action construct to indicate its dependency requirements on other tasks. The linear regression example given previously specifies a *disallow* dependency between the construction of the Three Group Regression Model (*constructTGRModel*) and the Parametric Significance Test (*parametricSignificanceTest*) technique. The states of KDPMEL tasks and their dependency requirements are recorded during the execution of the tasks. For example, when completing the execution of the *constructTGRModel* action, the following information is recorded for the action:

Action: *constructTGRModel* State: *Completed*

Dependency {Disallow: *parametricSignificanceTest*}

Upon beginning the execution of a task, a test is performed against the completed actions to check whether their *disallow* dependencies prohibit execution of the task. In the example above, when *parametricSignificanceTest* begins this conflict is found, the execution of the action is halted, and control proceeds to the next available action (*randomizationTest*). The *enable* dependency is checked only for the *choice* control construct to determine if one of the choices has been enabled by a completed action. If that was the case, the actor making the choice will be notified. Another test is performed upon beginning the execution of an action to check its requirement dependency (the *require* construct) against the completed tasks to determine if the action can be executed. An action can only be executed if its required tasks are completed successfully.

We believe that this is a novel approach for managing KDD task dependencies that are dynamically reflected at runtime, as opposed to statically structuring these tasks according to their dependencies at modeling time [3], which provides more flexibility not only in modeling time but in execution time also. In addition, it also leads to much shorter programs.

Action Specialized KDD Tasks

KDPMEL has a mechanism to model and execute environment-specific KDD tasks through the use of external commands that can be associated with an *action* and a *tool*. Each tool that is associated with an external command is represented by a plug-in that is invoked to execute the command in the surrounding environment. In the decision tree example given previously, the *build_C45_Weka_Classifier* command is executed by sending the command to the Weka plug-in to perform the execution of the command. This Weka plug-in translates the command into a valid Weka command and executes it. Fig. 4 illustrates this plug-in approach for external tool (Weka) commands.

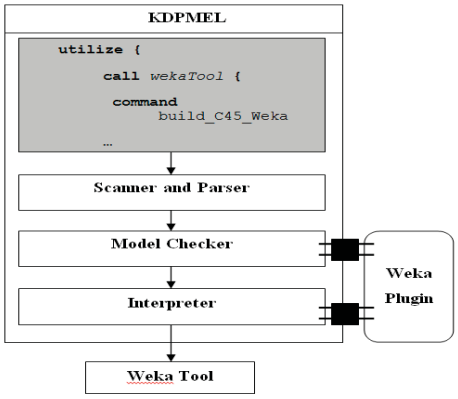


Fig. 4. KDPMEL External Commands Plug-in Mechanism

We believe that this is a novel approach that allows defining both generic and specialized KDD tasks at different levels of details and supports the goals of KDPMEL.

Task States and Transitions

The states and transitions of KDPMEL tasks are implemented using the State Pattern [14]. Tasks within a KDPMEL program go through several states during the execution of the program. The state of a task changes based on the control flow of the program, the availability of the resources needed by a task, and the explicit response from human actors. KDPMEL adopts states similar to those of Little-JIL--*posted*, *started*, *completed*, *terminated*, and *retracted*--- and adds the two new states *suspended* and *resumed* that have been suggested by Lee [5]. Fig. 5 illustrates KDPMEL task states and their transitions.

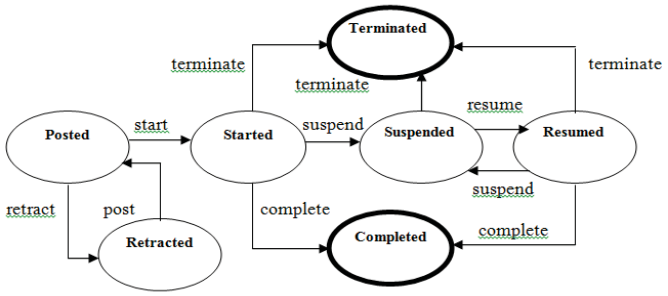


Fig. 5. Task State Transitions in KDPMEL [5]

When a KDPMEL task becomes available for execution, a task instance is created and its state is set to *posted*. A *posted* task instance is started by an explicit *start* action from the task performer (actor), which sends a start event to the task controller to change the state of the task instance to *started*. A *posted* task instance can also be temporarily retracted by a *retract* action.

A *started* task instance is completed by an explicit *complete* action. A *completed* task indicates that the task has successfully finished execution. This causes the enactment engine to continue executing the rest of the program by finding and posting the next available task. A *started* task instance can also be terminated by an explicit *terminate* action. A *terminated* task indicates an exception that caused the task not to be finished successfully. The handling of a *terminated* task varies depending on the type of control construct governing the task. A *terminated* task in a *sequence* or *loop* control construct causes the termination of the other tasks in the construct. A *terminated* task in a *parallel* control construct does not cause the termination of the other tasks in the construct. Therefore, the execution continues normally but the outcome of the terminated task is mainly affecting the status of its produced artifacts. Note that artifacts that are produced by a task are locked for exclusive use when the task is started and released when the task is finished with either *completed* or *terminated* state. A *terminated* task in a *choice* control construct causes the enactment engine to offer the construct alternatives, including the terminated task, to the actor to select a task. This approach promotes experimentation for performing KDD tasks, where an explicitly selected task can be tried first and maybe intentionally terminated later to either try another task or retry the same task if its outcomes were unsatisfactory. It also supports the interactive nature of KDD processes.

A *started* task instance can also be suspended by an explicit *suspend* action. A *suspended* task is handled in a way similar to a terminated one with some differences. In a *sequence* or *loop* control construct, the effect is the suspension of the other tasks in the construct. To continue execution, the suspended task must be *resumed*. In a *parallel* control construct, the execution continues normally but the effect of the suspended task is the continuation of locking its produced artifacts, thus preventing other tasks from starting execution because their needed artifacts are not released. This is different than a *terminated* task, where the lock is released so another task that needs to update the artifacts does not have to wait and can start execution. A *suspended* task in a *choice* control construct has the same effect of a *terminated* task and is handled similarly with the exception that it cannot be retried, because its needed artifacts are locked by the first try.

In addition to the explicit response from the actor to change the state of a task, other factors are considered before making the transition and changing the task state, such as the availability of the artifacts consumed by a task and the fulfillment of its *pre-conditions* before changing its state to

started. Similarly, the fulfillment of the task's *post-conditions* is required before changing its state to *completed*. Another factor is the control flow of the tasks and how each control construct affects the state of its tasks. Starting a *parallel* control construct causes the states of all its tasks to be changed to *posted*. Starting a *choice* control construct causes the state of the selected task to be changed to *posted*. Starting a *sequence* control construct causes the state of the first task in the sequence to be changed to *posted*. Subsequently, the state of the next task in the sequence is changed to *posted* only if the state of current task is changed to *completed*, either from *started* or *resumed*. A *loop* control construct is handled similarly with the difference that the state of the first task in the loop after the first iteration is changed to *posted* if the requirements of the first task following the loop are not met.

3 The KDD Process-Centered Support Environment (PCSE-KDD)

PCSE-KDD is an Integrated Development Environment that is built around KDPMEL, with an IDE-style approach to facilitate the development, execution, and management of KDPMEL programs. The environment aims to provide effective management for the KDD process by supporting its entire lifecycle, and offering a variety of services, similar to those offered by PCSEEs, but directed toward KDD and data analysis processes. Environment support includes assistance for developers, maintenance of process resources, automation of routine tasks, invocation and control of development tools, and enforcement of mandatory rules and practices. Fig. 6 illustrates the high level architecture of the environment. For additional information about PCSE-KDD, see [1].

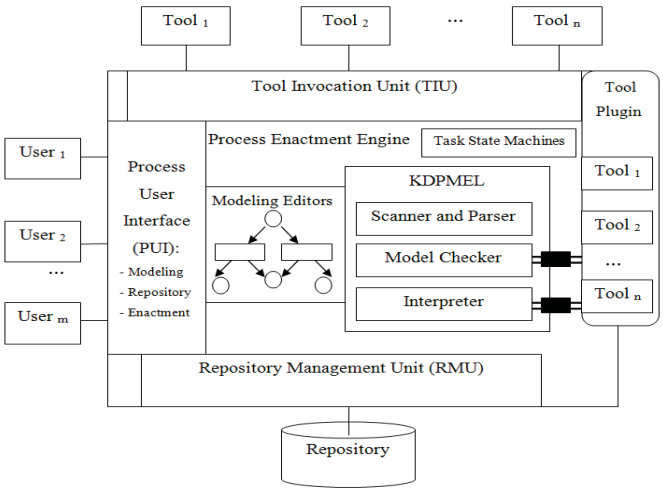


Fig. 6. The high level architecture of the PCSE-KDD

4 Evaluation and Lessons Learned

Our experience using KDPMEL to specify various aspects and tasks of KDD process supports our first hypothesis that a language-based and process-oriented approach is a flexible and effective approach to precisely and explicitly specify KDD processes as process programs that can be manipulated by programming techniques to reason about the process and support its correct execution.

KDPMEL simplicity goal achieved by having simple syntax that includes three major constructs for specifying the process resources (*artifacts*, *roles/actors*, and *tools*) and three major constructs for specifying the process tasks (*activity*, *action*, and *command*). The Language ease of use is supported by graph-based and form-based approaches to specify the various components of the process program in addition to the main source-based approach. The flexibility goal is mainly achieved by providing various levels for representing tasks at different levels of detail. Process tasks can be grouped using both strict and loose control to satisfy the highly variable control requirements of KDD processes. KDPMEL dependency control constructs allow controlling task dependencies at runtime, as opposed to static structuring in the program. The generality goal is mainly achieved by not making KDPMEL and PCSE-KDD bound to any particular KDD process models, techniques, or tools. This is accomplished by utilizing process technology to separate the KDD process definitions, which are represented as process programs, from their supporting environment, and by providing mechanisms for integrating these definitions with the environment using various programming techniques such as interpretation and validation. In addition, KDPMEL allows for specializing KDD tasks using the *command* construct and a plug-in mechanism that expand the environment with arbitrary external tools. The expressiveness goal is achieved by providing various constructs in KDPMEL to accurately reflect the KDD process by representing the tasks and their sequencing and dependencies, the artifacts consumed and/or produced by the tasks, the tools utilized by the tasks, and the actors performing the tasks.

During the implementation of a non-trivial KDD process [1], we were able to describe the process tasks and resources at different levels of detail because KDPMEL allows for specifying its constructs at arbitrary levels of detail. We were also able to execute the same process with minor variations to mine over different subsets of the data and to experiment with different tasks.

With regard to representing and integrating specialized KDD tasks, we have struggled between balancing the expressiveness and generality goals of KDPMEL. On one hand, the expressiveness goal suggests that the language should be able to accurately reflect KDD processes by representing both generic and concrete tasks, which requires representing particular KDD techniques and tools in KDPMEL. On the other hand, the generality goal suggests that the language should be applicable to KDD processes regardless of their particular models, techniques, and tools. The approach that we have adopted is to use a generic *command* construct along with a plug-in mechanism to translate these commands into representations that are accepted by concrete tools. We believe that this approach is quite general and supports all the language goals.

With regard to representing interactive KDD tasks that require the involvement of the task performer during the execution of the task, it was sufficient to represent these tasks using the KDPMEL generic *action* construct together with its associated *guidance* construct, which specifies informative guidance to help in carrying out the execution of the task.

Our experience using KDPMEL and PCSE-KDD to represent and execute various aspects of KDD process supports our second hypothesis that effective support and customized guidance, which depend on the concrete process itself rather than its generic process model, can be achieved by manipulating the explicit representation of the process in order to manage its various components and support its performance.

5 Conclusions

In this paper, we proposed the Knowledge Discovery Process Modeling and Enacting Language (KDPMEL) and outlined its Process-Centered Support Environment (PCSE-KDD) that can be used to develop KDD processes in a way that is similar to developing software processes, which is based on encoding KDD processes as process programs written in KDPMEL and exploited by PCSE-KDD to provide execution support and management for KDD processes.

KDPMEL provides a hybrid modeling approach for specifying KDD processes, mixing different types of editors and views in source-based, graph-based, and form-based styles to allow both technical and non-technical users to participate in the development of KDD processes. KDPMEL provides various language constructs to control task sequencing and dependencies as well artifacts consumed and/or produced, tools utilized, and the actors performing the tasks. KDPMEL allows for capturing the process tasks at different levels of abstraction to represent the process phases along with its generic and specialized KDD tasks. KDPMEL provides special control constructs that can be associated with a task to indicate its dependency requirements. This allows for explicitly representing and effectively managing dependencies among KDD techniques. KDPMEL provides the ability to explicitly model and enforce requirements and acceptance criteria of KDD tasks, which is important for making the KDD process more understandable, evaluating its correctness, assuring its consistent execution, and validating its results.

In KDPMEL, the process concept is supported and enforced according to a specialized KDD process that includes specific tasks organized according to their sequencing, dependencies, and alternatives.

6 References

[1] Mansour, H. A., Duchamp, D., and Krapp, C.-A. *A Language-Based and Process-Oriented Approach for Supporting the Knowledge Discovery Processes*. In Proceedings of the 11th International Conference on Data Mining (DMIN'15) (pp. 107-115), July 2015.

[2] Colin Shearer. *The CRISP-DM Model: The New Blueprint for Data Mining*. Journal of Data Warehousing, Volume 5, Number 4, 2000.

[3] David Jensen et al. *Coordinating Agent Activities in Knowledge Discovery Processes*. Department of Computer Science, University of Massachusetts Amherst, 1999.

[4] Noll, J. and Scacchi, W. *Specifying process-oriented hypertext for organizational computing*. Journal of Network and Computer Applications (2001) 24, 39-61, 2001.

[5] Lee, H. *Evaluation of Little-JIL 1.0 with ISPW-6 Software Process Example*. Department of Computer Science, University of Massachusetts, Amherst, MA 01003, March 1999.

[6] L. Kurgan and P. Musilek. *A Survey of Knowledge Discovery and Data Mining Process Models*. Knowledge Engineering Review, 21(1), pp. 1-24, 2006.

[7] Marban, O., Mariscal, G., Menasalvas, E., and Segovia, J. *An Engineering Approach to Data Mining Projects*. Intelligent Data Engineering and Automated Learning – IDEAL 2007, LNCS 4881, pp. 578-588, 2007.

[8] Marban, O., Segovia, J., Menasalvas, E., and Fernndex-Baizn, C. *Toward data mining engineering: A software engineering approach*. Information Systems 34 (1), 2009.

[9] OMG, Inc. UML Infrastructure Specification. URL: http://www.omg.org/technology/documents/formal/uml.htm. Version 2.0, March, 2006.

[10] OMG, Inc. Common Warehouse Metamodel (CWM) Specification. URL: http://www.omg.org/technology/documents/formal/cwm.htm. Version 1.1, March 2003.

[11] Greg Nordstrom et al. *Metamodeling - Rapid design and evolution of domain-specific modeling environments*. IEEE Engineering of Computer Based Systems (ECBS), Nashville, TN, April 1999.

[12] DePaul University, Chicago, IL. Classification via Decision Trees in WEKA. URL: http://maya.cs.depaul.edu/classes/ect584/WEKA/classify.html

[13] University of Waikato, New Zealand. Weka 3: Data Mining Software in Java. URL: http://www.cs.waikato.ac.nz/ml/weka/. Version 3.6, 2010.

[14] The State Machine Compiler (SMC) Framework. URL: http://smc.sourceforge.net/

[15] The Visitor Design Pattern. URL: http://sourcemaking.com/design_patterns/visitor

[16] OMG, Inc. Software Process Engineering Metamodel Specification. URL: http://www.omg.org/technology/documents/formal/spem.htm. Version 1.1, January, 2005.

Detecting Change in News Feeds Using a Context Based Graph

Lenin Mookiah, William Eberle
Department of Computer Science,
Tennessee Technological University,
Cookeville, TN, United States.

Maitrayi Mondal
Sunworks Consultants Private Limited,
Haryana, India.

***Abstract*— News feeds have been utilized as a resource for extracting media context, and in particular for the discovery of unusual information within common news articles. In this paper, we present our research of interesting unusual and useful patterns within the context of different social issues. We build a temporal context-based graph using news articles from different news channels built around actions employed by different entities, as well as resources involved with particular social issues such as accidental fires, kidnappings, human trafficking, road accident victims, mining accidents, ebola virus, swine flu, structure failures, senior citizen and juvenile incidences, migrant boat accidents, and slavery, with particular attention being paid to events such as agitation and the passage of legislation. From each article, we extract information such as the organization name, numbers, etc., and build a temporal graph of news articles. We then use sentiment analysis to measure the *sentiment* of each sentence, which are then used as edge values in our graph. For *context*, we utilize the graph structural connections among verbs, organization names, numbers, etc. For *content*, we use the new word count for each article. We propose a graph-cut model that leverages context, content, and sentiment information, empirically evaluate our proposed method, and present results that improve upon baseline methods in terms of precision, recall, F1, and accuracy.**

Keywords:- Change Detection, News Graph Mining

I. Introduction

Proliferation of news channels on the web has introduced a wide range of diverse data. These news articles actively report on stories involving crimes, terrorist attacks, and security issues relevant to the general population. Communities at risk deal with issues such as children forced into labor, senior citizens traveling in high-risk urban crime zones, and senior citizens not having access to public restrooms. Dark heterogeneous data sources such as news feeds, html, pdf files, and tables provide various statistical information and expert opinion analysis on these issues. For example, an article on child workers reported that "Half of the 5.5 million working children in India are concentrated in five states: Bihar, Uttar Pradesh, Rajasthan, Madhya Pradesh and Maharashtra"(Source:timesofindia.com, 13 Jun 2015). These types of web data provide a rich and complex set of information and knowledge on societal issues, such as policies proposed by the government or the implementation of a new law, that need to be extracted in a meaningful way for knowledge representation.

News mining has been studied in a variety of contexts. Earlier work involved grouping related news items, the fusion of news articles, and the summarization of information from disparate sources. However, news mining within a specific context could be more useful for a *targeted group of users* based upon their interests - what is commonly referred to as *personalized context mining*. Specific targeted groups could be based on age, location, gender, physical attributes, etc., or a variety of aspects based upon one's own interests. For instance, one could be interested in community challenges such as corruption, or perhaps safety in public places such as on a beach, road, or at a railway station.

First, we will define the various types of changes that our approach attempts to discover. Section III presents our definition of what constitutes a news article, or generally, a document. Section IV describes our process of data collection and preparation. Section V presents our proposed graph topology, and the tool used to extract the information, followed by Section VI that discusses the ground truth and evaluation methods for our experiments. Section VII presents our proposed method. Section VIII presents our experimental setup for our proposed method and baseline comparisons, followed by experimental results in Section IX. We then conclude with related work, conclusions, and future work.

Supported by NSF grant IIS-13185957.

II. Change Detected

First, we need to define what we mean by a document where there is "change detected".

Definition of Change Detected. A document (article) might contain one or more sentences in it that catch the attention of policy makers and/or groups interested in a particular social issue. In other words, the document includes one or more sets of information particularly useful to policy-makers and interest groups. In order for a document to be useful, the information should *have mention of a solution or intervention, an account of large resource damages, and/or contextual information explaining the issue*. We will mark articles containing such information as *change detected* articles. In other words, the article has "changed" from being just a typical news story.

Three Types of Change Detected. There are three types of *change detected* articles we are interested in from the perspective of interest groups:

- *Solution-based Change Detected (SBCD):* If an article contains a sentence that mentions an intervention or solution, then the article is marked as change detected. For example, precautions against potholes in order to avoid an accident:

TABLE I
FEATURES FROM A SAMPLE DOCUMENT (ARTICLE).

Name	Value
url	http://timesofindia.indiatimes/city/patna/one-kidnapped-in-vaishali-dist/articleshow/164404.cms
body	HAJIPUR: One Haribansh Rai of Mohanpur village of Vaishali district was kidnapped from his house on Tuesday night allegedly by the Sabal Rai gang. According to police sources, the kidnapping of Haribansh, 45, was due to an old enmity. He was kidnapped when he was fast asleep in his house. Sabal Rai's gang had created a reign of terror in the diara areas of Vaishali district. The gang is believed to be behind the killings of several truck drivers and cleaners.
date	2015/3/5

  - "Potholes rile commuters in the city. But after the Brihanmumbai Municipal Corporation (BMC) took cognizance of the menace and launched its pothole-tracking mobile application for people to complain about potholes."(Source: ndtv.com)

- *Context-based Change Detected (CBCD):* If an article is rich in context, such as getting attention from public and policy makers, then the article is marked as changed. For example, only a few global warming articles mention expert opinion for possible reasons behind the event, and when they do, information provided is very detailed. In other words, rich context articles provide asymmetric information content on the corresponding topic. For example:
  - "As the sea area freezes and melts each year, shrinking to its lowest extent ever recorded, Professor Peter Wadhams of Cambridge University called it a global disaster now unfolding in northern latitudes, the guardian reported." (Source: timesofindia.com)
- *Resource-damage-based Change Detected (RBCD):* If an article contains a number of resource losses, such as through injuries, deaths, or revenue, that are higher than expected, then the article is marked as changed. For example, buildings that collapse due to a conflict or an earthquake, usually result in a significant amount of property and lives lost:
  - "Around 1,300 people have been killed in Ukraine's Separatist conflict."(Source: ndtv.com)

III. DOCUMENT DEFINITION

A news article can be represented as a document, where D represents a collection of documents. Each document (article) is denoted by $d_i \in D$. Each document contains three features: *url*, *datetime* and *body*. Specifically, $d_i = \{url, body, datetime\}$, where url contains a web link to an article, $body$ refers to the *textual content* of the article, and $datetime$ contains the date and time of the article. Table I shows an example of features from a sample document.

The goal is that for each document $d_i \in D$, our algorithm will mark the document as either changed detected or not.

TABLE II
DATA STATISTICS FROM OUR DATA SET

Newspaper	Archive years	Total News Articles	Articles mentioning 12 societal issues
The Hindu	2009-2015 & 2000-2005	36715	358
Times of India	2009-2015	1726674	7527
NDTV	2009-2015	189976	2118

IV. DATA COLLECTION AND PREPARATION

The following is how we collected and prepared the data.

A. Data Collection

First, we crawled the index page of yearly archive pages from three Indian news papers - The Hindu[1], Times of India[2], and NDTV[3] - extracting the list of *url*s from the archives. The urls also contain the title of the article embedded in it, as shown in Table I. We then used grep on each url to filter news articles based on 12 different societal issues using the following keywords: fire, traffic, kidnap, senior citizen, juvenile, mining, ebola, swine, migrant, slavery, collapse, and road accident. It should be noted that while this particular list of keywords was chosen somewhat arbitrarily, it was based upon feedback from experts in India who deemed these particular issues of the most importance. We did not use any lexical expansion on our keywords search filtering, but we did filter out irrelevant articles, which reduced our experimental data set down to 8,433 news articles. Data statistics is shown in Table. II.

B. Data Preparation

Second, we analyzed the news articles related to the 12 societal issues for possible changes. Articles that report uncommon incidents within the context of social issues and policy, are marked for change detection. Each article was read by human annotators and searched for one or more of the change detected types defined previously: solution-based, context-based, and/or resource-based. If the article provides a *solution* or intervention to a social problem, such as the discovery of the ebola virus, it is marked as change detected because of the solution-based impact. If an article mentions experts' opinions, such as expert opinion on the ebola outbreak, that is considered a *context-based* change. If the article mentions huge resource losses, such as mass human casualties due to ebola, it would be considered a *resource-based* change, and would be marked as change detected.

We also discovered a few out of context (noise) articles. For example, an article mentioning a "cease fire" is inappropriate for a fire accident, and thus is removed from consideration. In addition, a few articles might contain information appropriate for one or more of the different change detected types. For example, the article at [12] is marked for change detection based on both resource-based and solution-based impact because 30 people died in the stampede making it eligible for being considered as a resource-based change, and a process was initiated to correct the root cause of the stampede, thereby also providing a solution.

V. GRAPH TOPOLOGY

Input for our approach is a graph. An example of our proposed graph topology of news articles is shown in Figure 1. In order to create this graph, we used openNLP[4] to

[1] $www.thehindu.com$
[2] $www.timesofindia.com$
[3] $www.ndtv.com$
[4] www.opennlp.apache.org

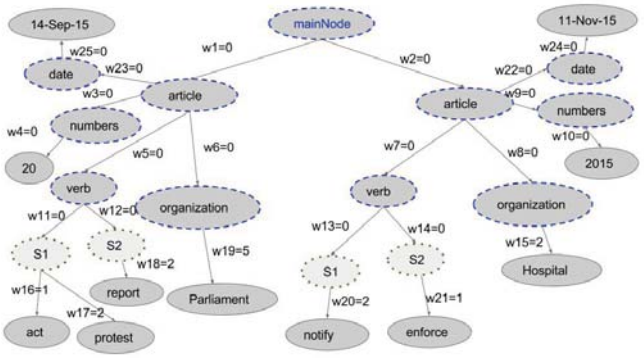


Fig. 1. Example graph topology of news articles.

extract the information doing part-of-speech tagging. For example, if a given sentence is "Cops arrested the murderer in the park", openNLP tags it as "Cops/NNS arrested/VBN the/DT murderer/NN in/IN the/DT park/NN". From this tagged sentence, we extract verbs, numbers, etc., as shown in Figure 1. We also used the Stanford NLP tool [5] to extract organizations as nodes, as shown in Figure 1 , where dashed and dotted lines represent types, such as "organization" and "verb", and nodes with solid lines represent actual values from the articles. For each article, we created an "article" node under a "mainNode" node (used solely for ease of retrieval). Under an "article" node, we created hard-coded number, verb, and organization nodes. For "verb", we further created hard-coded (dotted) nodes named $S1, S2, ..Sn$ where *S1* represents the first sentence, *S2* represents the second sentence, and so on. In other words, we create a tree-graph for each article. The extracted verb, numbers, and organization names, are attached as child nodes under their corresponding articles. The child nodes from one article can then be linked with other child nodes of other articles.

Graph weight. We use the Stanford Sentiment Analysis library[6] to mark the sentiment of each sentence in a news article. For each sentence, Stanford sentiment analysis predicts the following sentiment values: 1-very negative, 2-negative, 3-neutral, 4-positive, and 5-very positive. These values then represent edge weights in our graph. For all other edge relationships, we set it to *zero*. In Figure 1,w15, w16, w17, w18, w20, and w21 contain non-zero sentiment values. For all others that are zero, which we just represent as edge labels. The result is a weighted graph with vertices consisting of verbs, organization names, dates, and numbers. The resulting graph consists of *117,343* vertices and *231,142* edges.

VI. Evaluation

For the ground truth needed for evaluating our approach, each article is examined in detail by two policy making experts working for think-tanks. Annotator 1 is employed by *SunWorks Consultant Private Limited*, leading a team examining news article publications related to Chinese social polices, government activities, foreign policies, china neighborhood relations, and the economy. Annotator 2 also works in the area of Chinese affairs studies, employed at the *Institute of Chinese Studies (ICS)*. ICS is funded by *Ministry of External Affairs, Government of India*. ICS promotes interdisciplinary studies and research on China and the rest of East Asia with a focus on domestic politics, international relations, economy, history, health, education, border studies, language and culture. They crawled documents related to the three different change detected documents described previously, using the approaches described in the data preparation section, marking each article as change detected or not. It is important to note that only articles where both annotators would agree were marked as change detected. If there was a disagreement in terms of the context of a specific article, the article was removed from the data set. This resulted in 74 articles (out of the original 8,433) being marked as irrelevant.

In order to evaluate our approach, we use recall, F1-score, and accuracy, compared against existing standard approaches.

VII. Our Proposed Method

In this section, we propose an algorithm that uses an objective function based upon a greedy Graph-Cut approach. First, we present the important aspects relevant to change detected documents. Articles that are considered as appropriate for detecting this type of change involve one or more of the following aspects:

User-based Impact. An user is a named entity that is an organization, as marked by part-of-speech tagging (as discussed earlier). For example, articles of interest may mention an organization such as the World Health Organization (WHO). We then need to capture attribute information about the mentioned entity (user) particularly organization using *Stanford NLP* library. For example, the following sentence can be parsed to recognize that the entity: "The benefits of globalisation can be directed to reduce rural poverty if national and international economic policies take into account its effect on agriculture, according to Joachim von Braun, director-general, International Food Policy Research Institute (IFPRI)." (Source: thehindu.com, 23 Aug 2007). In this work, an *expert* is defined as the organization that is extracted using the *Stanford NLP* tool.

Action-based Impact. We are interested in capturing verbs that imply changes such as a new law being proposed or implemented. For example: "Delhi government clears Jan Lokpal Bill". The basic idea is to capture interesting verbs, such as "clears" or "passes", in the context of change. In this work, we try to leverage such verbs using graph structure.

Resource-based Impact. In the case of resource damages, lives lost (or hurt), or revenue lost, we need to capture attribute information about the resource affected. For example: "It was said at the time that over 6,000 houses were burnt." So, in this example, we want to capture the value 6000 in the context of houses that were destroyed.

In short, in order to classify an article as change detected or not, our method has to effectively capture and leverage contextual information that mentions user-based information (e.g., organization), action-based verbs (e.g., protest, strike),

[5] www.nlp.stanford.edu/
[6] www.nlp.stanford.edu/sentiment/

and resource-based information (e.g., "6000").

Graph G In this work, news articles are represented as a Graph G. Each verb, organization, and number, are represented as nodes. Edges connect nodes of the same value (verb, name, number) between news articles. D represents the set of articles (documents) in our data set. D_N represents the total number of documents in our data set. The definition of a news article is provided below.

Definition 1. An article. An article is represented as $d_i \in D$. Document d_i has N preceding documents by *datetime*. Each of the neighbors are represented by $d_j \in D$. D_N represents the total number of articles. A document d_i is defined as $d_i = \{d_i^{vb}, d_i^{org}, w_i^{neg}, N_{num}, N_{yr}, d_i^{neig}\}$. d_i^{vb} represents the number of common (action) verbs the article shares with all other articles in the graph G. d_i^{org} represents the number of organization names (tokens) mentioned. w_i^{neg} represents the number of negative sentiment sentences, N_{num} represents the number of times a number is mentioned (excluding numbers that represent a year), N_{yr} represents the number of times a year is mentioned. $d_i^{neig} = \{d_1, .., d_j\}$ represents the set of neighboring (preceding N) articles. N represents the number of preceding articles (neighbors) we wish to compare. We discover that a value of 5 gives an increased F1 score as shown in Table III, and is subsequently used as the minimum neighbor in our experiments.

Definition 2. Smoothing Function. In order to overcome noise while fitting the data for our model, we implement a smoothing function. We examined several smoothing functions, but most were sensitive to outliers such as zero. Thus, we ended up implementing equation 1. In particular, we require smoothing for features such as d_i^{vb} and d_i^{org}. We use fn_i^{vb} and fn_i^{org} as mentioned in equation 1 for d_i^{vb} and d_i^{org} respectively. First, this smoothing function provides smoothed values by converting outliers such as $d_i^{vb} = 0$, $d_i^{org} = 0$ to 1. Second, values are smoothed to be in the range [0, 1]. Precision and recall of the graph-cut with neighbor $N = 5$ are 0.4249 and 0.1675 respectively if we use d_i^{vb} and d_i^{org} instead of fn_i^{vb} and fn_i^{org} in our objective function. There is a decline in recall of 0.1675 from 0.4019 due to smoothing effect on new values that include outliers.

$$\begin{aligned} fn_i^{vb} &= fn(d_i^{vb}) = 1/(1 + (d_i^{vb})^2) \\ fn_i^{org} &= fn(d_i^{org}) = 1/(1 + (d_i^{org})^2) \end{aligned} \tag{1}$$

$$CC = \Big\{ \overbrace{\sqrt{fn_i^{vb}.fn_i^{org}} * w_i^{neg}}^{\text{A}} - \beta \overbrace{\frac{\sum_{j=1}^{N} \sqrt{fn_j^{vb}.fn_j^{org}} * w_j^{neg}}{N}}^{\text{B}} \Big\} * \underbrace{\{ d_i^{nword} - \frac{\sum_{j=1}^{N} d_j^{nword}}{D_N} \}}_{\text{C}} \tag{2}$$

$$\beta = \alpha * \{ \sqrt{N_i^{num} * N_i^{yr}} - \underbrace{\frac{\sum_{j=1}^{N} \sqrt{N_j^{num} * N_j^{yr}}}{N}}_{\text{E}} \} \tag{3}$$

Objective Function. The following discusses the context (graph structure) information, content information, and cut cost (CC) used in our proposed greedy approach. For content and context information, we remove the stop words and perform word-stemming. Our approach greedily calculates the minimum graph-cut in Equation 2 in comparison with its neighbors. We leverage two types of information from each article: Context and Content. Context leverages information such as verb, numbers, and sentiments using graph structure. Content leverages information such as new words and similarity metrics. Our algorithm is based on the strength of context and content of an article *with its neighbor*.

Context. Contextual information involves the mention of common action verbs (social context) and popular social organizations across multiple articles, thus forming edges, or links, across documents. For example, two or more articles might be connected via the node of a common verb such as "act", "notify", or "announce". These common (action) verbs might occur in phrases such as "Law passed" or "New traffic rules announced", indicating a change (per our definition). Similarly, organization names and numbers, such as the year in one document, can form linkages to other documents. For instance, the more numbers are mentioned in an article, the more likely it has a statistical significance. In addition, more numbers in an article likely indicates that resource-based changes are contained in the article. For example: "2100 died because of ebola in year 2015 alone". In short, we use organization names, numbers, and common action verb linkages, tagged using the *openNLP* library and *Stanford NLP* tool, for the contextual (structural) information in the graph.

Content. Content information that carries *new words* is considered to be carrying new information. For example: "New mobile application has been introduced for senior citizens.". Equation 3 is based upon new words. For each document d_i, we calculate the count of new words that have not occurred in *N* preceding documents, which we call *neighbors*. Thus, in terms of context and content, our first intuition is that an article with considerable changes will also mention the organization name, common (action) verbs, numbers, etc. Our second intuition is that this article will have a fewer number of negative sentiments than its neighbors. This is due to fact that these change detected articles might have information such as solutions proposed, statistical information, and guidelines mentioned by experts - all (presumably) positive information to the reader. In other words, interesting change detected articles might have *fewer* sentences with negative sentiments. In other words, an interesting change detected article will have fewer w_i, and is more than likely mentioning numbers, organization names, and action verbs, as compared to its neighbors.

Cut Cost. For each article, we calculate Cut Cost (CC), as

defined in Equation 2. We first calculate the geometric mean of the common verb function fn_i^{vb} and the organization function fn_i^{org} for a given document d_i, which is marked as *Part A* in equation 2. Then, the geometric mean is compared to the average of its neighbors, which is marked as *B* in equation 2. Additionally, the count of the number of negative sentences in the article w_i^{neg} is multiplied against the geometric mean. Similarly, for a given document d_i, we calculate a d_i^{nword} count of new words that have not occurred in the past *N* documents. *Part C* of equation 2 shows where we capture the difference of the calculated new words of d_i with the average number of new words of all documents in the data set. If the cut cost is greater than zero, we mark the article as change detected. Again, the basic intuition is that a change detected article will have mix of statistics, less negative sentiments, and organization names that act as change agents in comparison to their neighbors.

Penalty For articles that contain no relevant information on context, our objective function needs to over-penalize them on the cut cost. To do this, we introduce penalty β as shown in Equation 3. β captures the difference between the geometric mean of the count of numbers and the count of years, against the average of its neighbors. *Part E* of Equation 3 represents the average of its neighbors' geometric mean, count of numbers, and count of years. Parameter α controls the degree of penalization. In our experiments, we evaluated $alpha$ from 0.1 to 40 and found the best $F1$ to be when $\alpha = 0.95$. Figure 2 shows the effect of α on precision, recall, and F1. After $\alpha = 2$, precision, recall and F1 do not improve, and actually maintain the same percentage.

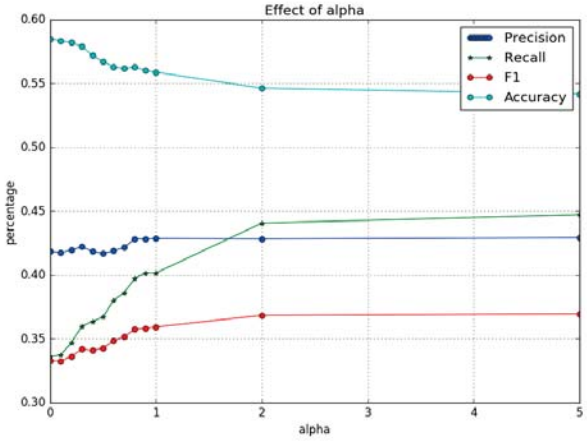


Fig. 2. Effect of penalty parameter alpha.

VIII. Experiment

The four comparison methods we use are cosine similarity, new word count, new word count with threshold, and jaccard coefficient. In our comparison methods, as mentioned earlier, for all *N* preceding document comparisons we use a value of 5 for *N*. All experiments are run on a Mac with 2.8 GHz Intel Core i7, with 16GB of memory, and a 1600 MHz DDR3.

A. Our proposed Method.

We implement our proposed method as an iterative algorithm, as shown in Algorithm 1. The algorithm iterates over one document d_i at a time. It gets all the child nodes such as date, number, verb, and organization under the "article" node of d_i, then calculates the cut cost for d_i as per Equation. 2. If cut cost is greater than 0, it classifies it as "change detected".

Algorithm 1 Graph-Cut Algorithm

```
Input: D documents of our data set.
alpha = 0.9
procedure GRAPH–CUT–ALGORITHM
    for each article d_i ∈ D get "article" node do
        Get all child nodes under "article" node of d_i
        Calculate cut cost CC for d_i as per Equ. 2
        IF calculated CC > 0
            classify d_i as "change detected"
        ELSE classify d_i as "NOT change detected"
    end for
end procedure
```

B. Comparison Methods

Cosine similarity [13] and *Jaccard Coefficient* [14] are the two most popularly used methods. These methods have also been applied to problems such as novelty detection [4]. We use them to calculate TF-IDF similarity between 2 articles. Another approach involves calculating the *new words* count when comparing a document to past documents, which can be used to discover an uncommon document. We also chose this technique as one of our baselines as it has been used repeatedly in other related research [6][7][9]. For our experiments, the articles from all 3 news papers listed in Table II are merged and sorted chronologically. Since several articles from *thehindu* newspaper have missing dates, we only use the month and year for the chronological ordering of news articles. Our basic intuition is that rich (uncommon) contextual documents will have less similarity with their recent past and future documents. For each of the baseline methods, D^q represents a set of documents chronologically ordered containing a topic query q . We prepared a TF-IDF for each document $d_i^q \in D_q$. TF-IDF methods are often used as a fast and effective means of comparison with similarity based methods such as cosine similarity. We also removed stop words and performed word-stemming.

Cosine Similarity For a query q, we iterate through each individual document d_i^q and calculate the cosine similarity for each of the N preceding documents. Then, we average the pair of documents' calculated cosine similarity. The Top n% of documents with the least cosine similarity are marked as change detected. As discussed under *Definition 1*, for *N* preceding documents in the range [1-4]. Thus, we experimented with values of 5, 10, and 15, and found the best precision, recall, and F1-score using $N = 5$.

New Word Count. For a query q, we iterate through each individual document d_i^q and count the new words that have not occurred in each of the *N* preceding documents. Then, we average the new word count for all pairs of documents. The Top n% of documents with the highest new word count are marked as change detected.

New Word Count with Threshold. This method is similar to the *new word count* approach. In this case, we provide a threshold of percentage difference between new words found in an individual document d_i^q with the average new word count of *N* preceding documents. The Top n% of documents with the highest percentage difference are marked as change detected. We experimented with threshold values between 80 and 90, and discovered the best precision using a $threshold = 90$.

TABLE III

RESULTS SHOWING PRECISION (PREC), RECALL, F1-SCORE, AND ACCURACY (ACC) FOR BASELINES METHODS AND OUR GRAPH-CUT APPROACH. FOR BASELINE METHODS, THE TOP 20% OF ARTICLES ARE MARKED AS CHANGE DETECTED. DIFFERENT VALUES FOR *N* REPRESENTING NUMBER OF PRECEDING DOCUMENT (NEIGHBORS) EXPERIMENTS AND RESULTS SHOWN.

Top n%	N	Method	Prec	Recall	F1-score	Acc
20%	5	Cosine	0.3462	0.1869	0.2064	0.5489
20%	5	New word count	0.4207	0.1799	0.2363	0.58
20%	5	New word count-Threshold	0.2363	0.2935	0.1778	0.6414
20%	5	Jaccard	0.355	0.1768	0.2112	0.6648
-	5	Graph-Cut	0.4283	0.4019	0.3583	0.5589
20%	10	Cosine	0.3363	0.1831	0.1979	0.5451
20%	10	New word count	0.429	0.1857	0.2426	0.5832
20%	10	New word count-Threshold	0.2363	0.2935	0.1778	0.6414
20%	10	Jaccard	0.3673	0.1824	0.2173	0.6672
-	10	Graph-Cut	0.4266	0.4	0.3565	0.5576
20%	15	Cosine	0.3231	0.1767	0.1912	0.5399
20%	15	New word count	0.4256	0.1842	0.2407	0.5818
20%	15	New word count-Threshold	0.2363	0.2935	0.1778	0.6414
20%	15	Jaccard	0.3802	0.188	0.2246	0.6698
-	15	Graph-Cut	0.4289	0.4055	0.3611	0.5611

Jaccard Coefficient. For a query q, we iterate through each individual document d_i^q and calculate the jaccard coefficient for each of the *N* preceding documents. Then, we average the pair of documents' calculated jaccard coefficients. The Top n% of documents with the lowest jaccard coefficients are marked as change detected.

In all above methods, we iterate through each document from a chronologically sorted set. We usually take *N* preceding documents. However, for the first few documents, there might not be any preceding documents. Hence, we take either the *N* succeeding documents (if available), or a combination of preceding and succeeding documents. For example, when iterating on the 6^{th} document, we need a total of *N*=15 preceding documents, where as we got only 5 preceding ones. In this case, we then include the succeeding 10 documents.

TABLE IV

RESULTS SHOWING EFFECT OF DIFFERENT *Top n %* VALUES MARKED AS CHANGE DETECTED ON PERFORMANCE OF BASELINES.

Top n%	N	Method	Prec	Recall	F1-score	Acc
10%	5	Cosine	0.3070	0.0842	0.1102	0.5727
10%	5	New word count	0.4692	0.0923	0.1471	0.6040
10%	5	New word count-Threshold	0.2021	0.1640	0.1169	0.6230
10%	5	Jaccard	0.4042	0.0933	0.1384	0.6420
15%	5	Cosine	0.3445	0.1322	0.1566	0.5605
15%	5	New word count	0.4729	0.1371	0.1974	0.5971
15%	5	New word count-Threshold	0.1839	0.2240	0.1349	0.6262
15%	5	Jaccard	0.4103	0.1357	0.1810	0.6540

IX. RESULTS AND DISCUSSION

Table III shows our experimental results, comparing baseline methods to our proposed Graph-Cut approach. We first compare and discuss results when dealing with the Top 20% and $N = 5$. The accuracy of the baseline approaches, except *cosine similarity*, are noticeably better than our Graph-Cut approach. However, Graph-Cut gives the overall better precision (*new word count* comes close), recall, and F1-score. We also varied values of *N* preceding (neighbor) documents, with results for *N* values of 10 and 15 shown in Table III. The results are similar to that of *N* with value 5, albeit the *new word count* approach again is close or even slightly better in terms of precision, but not better when it comes to recall or F1-score.

In addition, we evaluated the baseline methods with different values for the *Top n %* other than the *Top 20%* marked for change detection. It is worth to note that the percentage of change detected articles in our data set is approximately 14%, so we experimented with values of the Top 10% and Top 15% being marked as change detected. We include Table IV to further show that results are only impacted by different values of n used for the *Top n %* marked as change detected documents. For the Top 15 %, performance reduces in precision, recall, and F1, and for the Top 10% performances is even lower. In Fig. 2, graph-cut achieves the best $F1$ when $\alpha = 0.95$. After $\alpha = 2$, all of our measures flatten out. For Top 15% and N=5, student's t-test of our 4 evaluation metrics of New word count with Threshold and Graph-Cut shows significance only at 14 %. However, Graph-Cut has required better F1 and accuracy.

It should also be noted that the running time of the baselines algorithms ranges anywhere from 5-10 seconds. In comparison, our Graph-Cut method takes approximately 1 second to complete.

X. RELATED WORK

Our work is most closely related to the research that is being done on novelty detection, particularly in a temporal setting. Gaughan and Smeaton [3] study novelty detection

using the TREC data set. The NIST TREC[7] data is from 2002-2005, and includes tracks that are divided into event and opinion topics. For example, they use the 2004 data, which uses 3 news feeds from Xinhua, the New York Times and the Associated Press. In their study of novelty detection, they employ a Term Frequency-Inverse Document Frequency (TD-IDF) variant. The authors use *F1-score* for evaluation, and achieve an F-score of 0.622 and 0.807 on the 2004 and 2003 data respectively. Li et al [6] [7] also study the novelty detection problem using TREC. First, their algorithm converts the query into a query and its expected related answer type. The basic idea is that if there is a combination of query words, named entities, etc., available in a sentence, this increases the possibility of answer. The approach uses a concept called "answer patterns". Answer patterns are a list of answer candidates, each with a specific pattern prepared for each question using a belief or heuristic .

Schiffman et al. [8] leverage contextual information for the novelty detection problem. The authors use the context of the sentence along with novel words and named entities. The algorithm tries to find the optimal value for 11 parameters, weights, and thresholds. The algorithm uses a random hill-climbing algorithm with backtracking for learning weights. The algorithm achieves a recall of 0.86 on the average of all runs, in comparison to cosine similarity with 0.81. Karkali et al. [4] study the problem of online novelty detection on news streams. Their work uses two data sets, one from the Google news RSS feed and another from Twitter. Novelty is defined in terms of a predefined window on the past. The proposed algorithm is based on the TF-IDF , and is evaluated using a linearly combined single detection cost [5].

Our work differs from previous efforts as our proposed method is a graph-based approach. Our graph-based approach works better because of advantages in leveraging (1) structure, (2) content information, and (3) context information. We use an aspect-level and fine-grained approach of individually extracting and using dates, numbers, organization, and (4) sentiments in graph. Also, we are currently focused on discovering news articles relevant to policy makers - a problem for those who are tasked with generating relevant societal policies.

XI. Conclusion

In this study, we collected data from 3 different news sources. We extracted common verbs, organization, and numbers, and built a weighted graph using sentiments of each sentence in news articles. We proposed a greedy graph cut algorithm that outperforms baselines in precision, recall, and F1. We also study the penalty parameter α and report the impact on our evaluation metrics. In the future, we will investigate using an external ontology for policy making such that it could help to better leverage the context (structural informal). For example, in wikipedia[8] the "Union Council of Ministers of India" provides different department names within the government. This might enable us to better capture entities such as organization name. In addition, we will examine augmenting the graph with certain important features such as the designation of people names. For example: "Forensic science officials will also be called upon to examine the building quality". One challenge here is to effectively extract the designation (i.e., "forensic science officials"). Due to an overwhelming number of common nouns, these designations (nouns) become underrepresented and hence are not used effectively. Also, we are currently working on extending our temporal graph to graph streaming approaches, thereby exploring related streaming techniques.

[7] www.trec.nist.gov

[8] https://www.wikipedia.org/

Acknowledgment

We sincerely thank Jayshree Borah from the China Studies Centre, Indian Institute of Technology Madras for helping in labelling the articles, and providing useful feedback. This material is based upon work supported by the National Science Foundation under Grant No. 1318957.

References

[1] Yu W, Aggarwal CC, Ma S, Wang H. *On anomalous hotspot discovery in graph streams*, In Data Mining (ICDM), IEEE 13th International Conference (2013), pp. 1271-1276.

[2] Arackaparambil C, Yan G. *Wiki-watchdog: Anomaly detection in Wikipedia through a distributional lens*, In Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology-Volume 01, (2011), pp. 257-264.

[3] Gaughan G, Smeaton AF. *Finding new news: Novelty detection in broadcast news*, InInformation Retrieval Technology (2005), pp. 583-588.

[4] Karkali M, Rousseau F, Ntoulas A, Vazirgiannis M. *Efficient online novelty detection in news streams*, In Web Information Systems Engineering-WISE (2013), pp. 57-71.

[5] Manmatha R, Feng A, Allan J. *A critical examination of TDT's cost function*, In Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval (2002), pp. 403-404).

[6] Li X, Croft WB. *Novelty detection based on sentence level patterns*, In Proceedings of the 14th ACM international conference on Information and knowledge management (2005), pp. 744-751.

[7] Li X, Croft WB. *Improving novelty detection for general topics using sentence level information patterns*, In Proceedings of the 15th ACM international conference on Information and knowledge management (2006), pp. 238-247.

[8] Schiffman B, McKeown KR. *Context and learning in novelty detection*, In Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing (2005) pp. 716-723.

[9] Li X, Croft WB. *An information-pattern-based approach to novelty detection*, Information Processing & Management, 44.3, (2008), pp. 1159-1188.

[10] Shahaf D, Guestrin C. *Connecting the dots between news articles*, In Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, (2010), pp. 623-632.

[11] A. N. Michel and R. K. Miller, *Qualitative Analysis of Large Scale Dynamical Systems*, New York: Academic Press, 1977.

[12] http://ndtv.com/allahabad-news/allahabad-stampede-not-due-to-railing-collapse-railway-minister-pawan-kumar-bansal-512954

[13] Blank D. *Resource Description and Selection for Similarity Search in Metric Spaces. University of Bamberg Press; 2015 May 7.*

[14] Goodman LA, Thompson KM, Weinfurt K, Corl S, Acker P, Mueser KT, Rosenberg SD. *Reliability of reports of violent victimization and posttraumatic stress disorder among men and women with serious mental illness. Journal of traumatic stress. 1999 Oct 1;12(4):587-99.*

PCSE-KDD: A Process-Centered Support Environment for the Knowledge Discovery Processes

Hesham A. Mansour
FJA-US, Inc., 1040 Avenue of the Americas, 4th Floor, New York, NY 10018, USA

Abstract – *Current support for Knowledge Discovery in Databases (KDD) is provided only for fragments of the process, a particular KDD process model, or most recently certain process aspects. The support needed for a KDD process varies greatly based on the specifications of the concrete KDD process, and cannot be based purely on a generic process model. There is a need for a more comprehensive support approach that can cover the entire process, target concrete process specifications, and include various aspects of the process. KDD processes are similar to software processes and they can benefit from advancement of software engineering and process technology to facilitate their development, support their execution, and ultimately improve their effectiveness, utilization, and outcomes. This paper proposes the Process-Centered Support Environment for KDD (PCSE-KDD) processes that is based on explicitly representing these processes as process programs that can be developed, managed, and enacted by the environment. This approach has been successfully used to provide support for developing software processes and we propose to transplant this approach into the KDD field. With the proposed approach, KDD processes can be flexibly captured at different levels of details in a clear, precise, and explicit way that can enable reasoning about the process, insuring its correct execution, and supporting its performance.*

Keywords: KDD Process, Process Programming, Process-Centered Support Environments

1 Introduction

Although KDD is now widely accepted as a complex process with many different phases and non-trivial interactions, little support is provided to the various steps of the process or to manage the overall process.

The lack of systematic approaches for managing and keeping track of the different parts of KDD projects means that some steps may unintentionally be repeated, adding overhead to the knowledge discovery task. Rudiger et al. [18] have noted major problems during the development of many KDD projects at Daimler-Benz due to the lack of a methodology and lack of a usable process model with proper tool support. The result is wasted resources and unnecessarily long development times, in addition to the fact that the results were highly dependent on the experience of the persons doing the work. Marban et al. [8] have noted that the number and complexity of data mining projects has increased in recent years, that nowadays there isn't a formal process model for this kind of project, and that existing approaches are not correct or complete enough. They also noted that not all projects end successfully. The failure rate is actually as high as 60%. The intrinsic features of the KDD process, together with the main difficulties in its application, make the development and management of a KDD application, particularly of exploratory nature, very complex [19].

Current support for KDD is provided only for fragments of the process (*activity-oriented* support), a particular KDD process model (*KDD support environments*), or most recently certain process aspects (*process-oriented* support), such as the coordination or collaboration [1]. In the activity-oriented support approach, the process concept, if used at all, is only represented in the form of documentation and guidelines. Also, the tools supporting the process tasks are isolated without any means of integration. The process support provided by most existing KDD support environments is mainly derived from a hardwired generic KDD process model, which includes major process phases along with their generic tasks and simple interactions. This sort of guidance is too generic and clearly insufficient for effectively supporting KDD processes, where specialized guidance is needed to assist in selecting valid, desirable, and effective process configurations. Moreover, the tool guidance provided by these systems is limited to a few standard KDD techniques and prescribed set of supporting tools that are mandated by the environments. Among the very few proposals that apply process-oriented support to KDD, only [4] uses a process language approach based on Little-JIL to explicitly represent and support only the coordination aspect of KDD processes. In addition to the discovered deficiencies in Little-JIL, only the simplest processes can be modeled visually using Little-JIL. For additional information about the different approaches for supporting KDD processes and their limitations, see [1].

The recognition that software processes can themselves be described as software is attributed to Osterweil [20], and has led to the development of process programming as part of software engineering, as well as ongoing research into process-centered environments. The idea of using a Process

ISBN: 1-60132-431-6, CSREA Press ©

Modeling Language (PML) to encode a software process as a "process model", and enacting this using a process-sensitive environment is now well established [21]. Process-Centered Software Engineering Environments (PCSEEs) form the most recent generation of environments supporting software development activities [22]. They aim to support software development activities by exploiting an explicit representation of the software process---a process program---that specifies how to carry out the process activities and how to use and control the process supporting tools.

Although some researchers [4], [7]-[9] have recognized the similarities between KDD processes and software development processes, none (to our knowledge) has proposed a comprehensive approach for developing KDD processes through a Process-Centered KDD Support Environment based on PCSEEs to enable and facilitate the modeling, execution, and management of KDD processes.

In this paper, we propose the Process-Centered Support Environment for KDD (PCSE-KDD) for modeling, enacting, and managing KDD processes. The environment aims to provide effective management for the KDD process by supporting its entire lifecycle, and offering a variety of services, similar to those offered by PCSEEs, but directed toward KDD processes. Environment support includes assistance for process developers, maintenance of process resources, automation of routine tasks, invocation and control of development tools, and enforcement of mandatory rules and practices. The environment implements the process definition/instantiation/enactment paradigm found in PCSEEs and is based on the KDD process programming language KDPMEL [1], [2]. The environment includes a number of modeling editors for modeling KDD processes, an Enactment Engine for providing runtime process execution support, and a Repository for providing persistency support to both process artifacts and process execution states.

Achieving a general-purpose data mining and knowledge discovery support environment is an undertaking that has been described to be quite a challenging problem in [23] and was predicted in 2003 to be among the most important KDD issues that will not show any measurable and notable scientific progress in the next 10 years [24]. This pessimism was mainly because of the complex nature of the KDD process and its branching factors in terms of selecting specific methods and supporting tools, branching which has caused commercial data mining products to be limited to a few standard techniques and to provide guidance based only on a hardwired process model. In contrast, we demonstrate that a general-purpose KDD support environment can be achieved by separating KDD process definitions from the environment and by providing the appropriate mechanisms for integrating these definitions with the environment. This separation of concerns can achieve significant flexibility in supporting a wide range of process specifications that can evolve over time and generality due to the fact that the environment is not bound to any particular KDD process models, techniques, or tools. Process technology can provide the appropriate approaches for achieving this separation of concerns. The paper is structured as follows. This section provides background information and the motivation for our work. Section 2 presents the Process-Centered Support Environment for KDD (PCSE-KDD) and illustrates its major components. Section 3 outlines the implementation details of PCSE-KDD. Section 4 concludes the paper and outlines future work.

1.1 Process-Centered Software Engineering Environments (PCSEEs)

PCSEEs address three distinguishable domains: the modeling, enactment, and performance domains. The modeling domain comprises all activities for defining and maintaining process models using a formal language with an underlying operational semantics that enables mechanical interpretation of the models. The enactment domain encompasses what takes place in the environment to mechanically interpret the process model by a so-called process engine. The performance domain is defined as the set of actual activities conducted by human agents and nonhuman agents (computers) during process execution. Process support provided by PCSEEs can be characterized by the typical interactions between the three domains (Fig. 1) [10]:

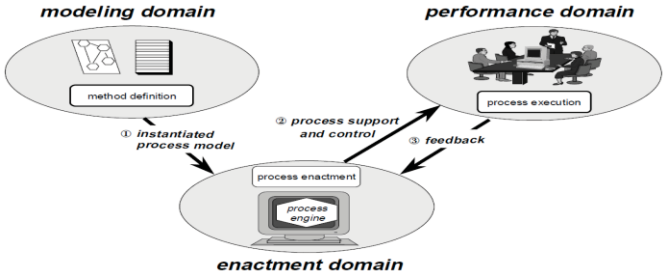


Fig. 1. Three domains of software process support [10]

1.2 KDD Processes and Software Development Processes

KDD processes are on one hand similar to software processes and on the other hand are different from software processes. The similarities between KDD processes and software processes suggest that approaches used to support the development of software processes, such as PMLs and related PCSEEs, are also applicable to KDD processes. However, because of the differences, some adaptation is needed in order to apply these approaches to KDD processes. Instead of adapting current KDD processes to match software processes as proposed in [8], [25], we propose to adapt these approaches to suit KDD processes in order to support their specific activities, techniques, components, and developers. This will provide support for current KDD processes as they are normally known by KDD practitioners who are not necessarily experts in software engineering. In addition, this will not force fundamental changes and additional activities on KDD processes and at the same time will not prevent form doing so when needed.

The approach that we propose to establish formal process models and methodologies for developing KDD processes is based on transplanting the idea that has been successfully used in software engineering to support the development of software processes into the KDD field. By transplanting this idea to KDD, we believe that we can formally and explicitly define KDD processes and provide a systematic methodology for their development and execution.

2 The KDD Process-Centered Support Environment (PCSE-KDD)

PCSE-KDD is an Integrated Development Environment that is built around KDPMEL, with an IDE-style approach to facilitate the development, execution, and management of KDPMEL programs. KDPMEL provides a hybrid modeling approach for specifying KDD processes, mixing different types of editors and views in source-based, graph-based, and form-based styles to allow both technical and non-technical users to participate in the development of KDD processes. KDPMEL provides various language constructs to control task sequencing and dependencies as well artifacts consumed and/or produced, tools utilized, and the actors performing the tasks. KDPMEL allows for capturing the process tasks at different levels of abstraction to represent the process phases (lifecycle) along with its generic and specialized KDD tasks. For additional information about KDPMEL, see [1], [2].

2.1 Architecture

Fig. 2 illustrates the high level architecture of the environment.

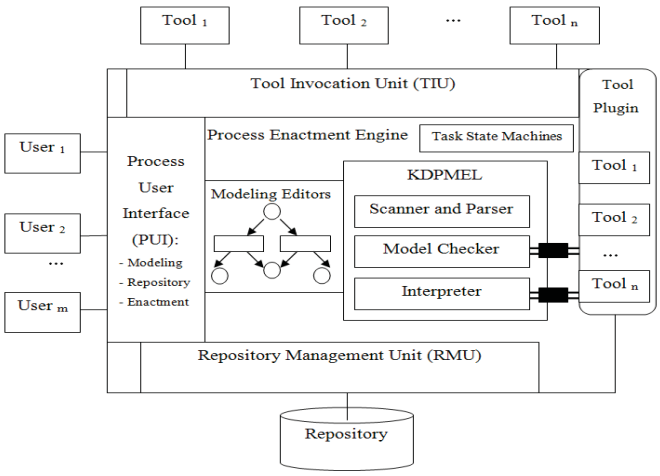


Fig. 2. The high level architecture of the PCSE-KDD

The PUI exposes the various components and services offered by the environment. Through the PUI, users are able to define, update, and persist process models/programs during the modeling phase, instantiate a process model for enactment, participate in the enactment phase by performing manual and/or interactive tasks in the process, are notified by the enactment engine about the status of the process being enacted, and are guided by the enactment engine about what to do next. The PUI uses the *perspective* concept in a way similar to Eclipse's perspectives [26] to control the visibility and presentation of items in the workspace of the environment. The PUI includes three different perspectives to support the modeling, enactment, and management features of PCSE-KDD.

The Enactment Engine includes three significant components: KDPMEL Interpreter, the Repository Management Unit (RMU), and the Tool Invocation Unit (TIU). The KDPMEL Interpreter implements the semantics of the language. The RMU maintains the process data during process modeling and enactment. The TIU manages the invocation of tools specified in the process program. Tools are specified in the resources section of the program and they can be referenced by the KDPMEL *action* construct. Two types of tools can be specified. The first type is interactive tools. The invocation of an interactive tool is based on a URL representing the tool executable. The second type is scripted tools that can be run from the KDPMEL *command* construct.

2.2 PCSE-KDD Perspectives

The PUI-Modeling perspective includes the items visible during the modeling phase along with their provided presentations and supported actions in the user interface. The PUI-Enactment includes the items relevant to the enactment phase. The PUI-Repository perspective includes the items maintained in the environment repository.

2.2.1 The PUI-Modeling Perspective

The PUI-Modeling perspective supports the three modeling approaches provided in KDPMEL: source-based, graph-based, and form-based. Fig. 3 sketches the layout of the PUI-Modeling perspective.

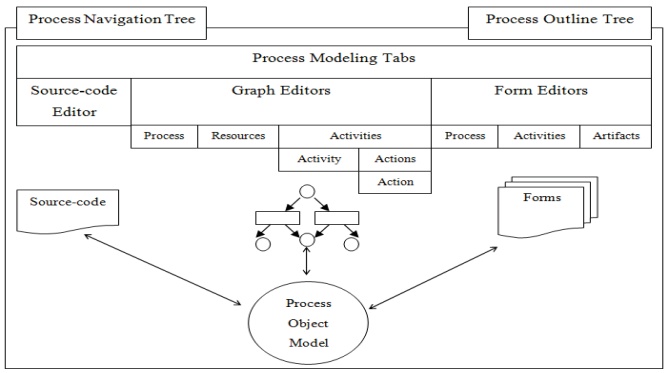


Fig. 3. The PUI-Modeling Perspective

The Process Navigation Tree allows for navigating between multiple processes. The Process Outline Tree displays the process components grouped by their types and allows for navigating between these components in the source-code representation of the process. The Process Modeling Tabs includes a tab for each process shown in the Process Navigation Tree. The Process tab includes a tab for the Source-code editor, a tab for the Graph editors, and a tab for the Form editors. The Graph Editors tab includes a tab for

the Process Graph editor, a tab for the Resources Graph editor, and a tab for the activities. The Activities tab includes a tab for each activity. The Activity tab includes a tab for the Activity Graph editor and a tab for the actions. The Actions tab includes a tab for each action. The Action tab includes the Action Graph editor. The Form Editors tab includes a tab for the Process Form editor, a tab for the activities, and a tab for the artifacts. The Activities tab includes a tab for each activity. The Artifacts tab includes a tab for each artifact. The Artifact tab includes the Artifact Form editor.

Fig. 4 depicts a view of the PUI-Modeling perspective showing its Source-code editor.

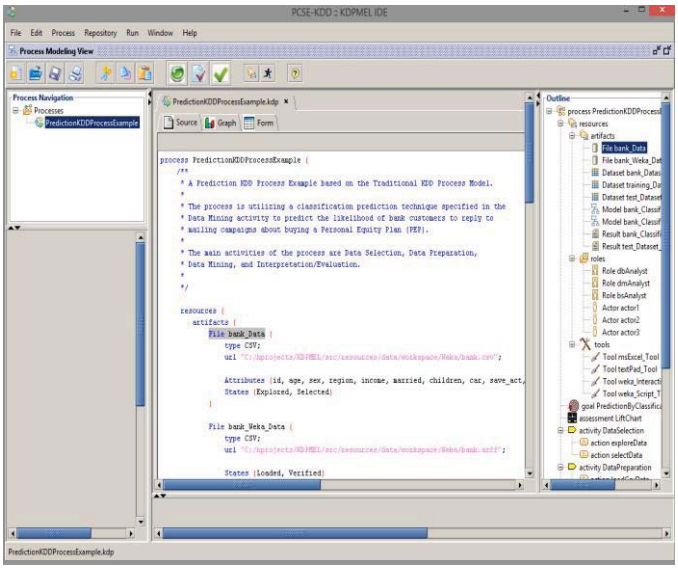


Fig. 4. The Source-code editor of the PUI-Modeling Perspective

A KDPMEL program can be developed using a Source-code editor and then updated using the Graph and Form editors that are automatically created from the program source-code [2]. The alternative is to use the various graph editors: the Process Graph editor to create the process and its activities; the Resources Graph editor for process resources; the Activity Graph editor for each activity along with its constituent actions; and the Action Graph editor for each action. Process graphs are translated into their source-code and form representations. Fig. 5 through Fig. 7 depict multiple views of the PUI-Modeling perspective showing some of its various Graph editors.

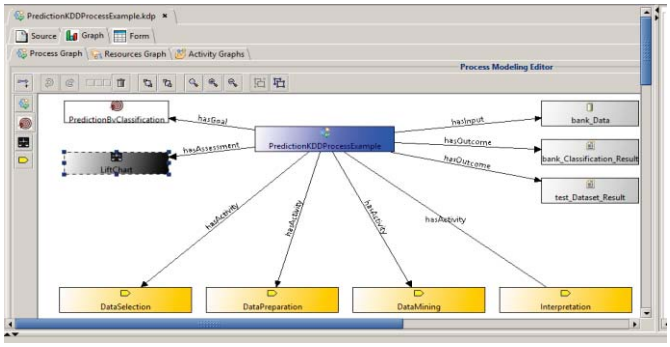


Fig. 5. The Process Graph editor of the PUI-Modeling Perspective

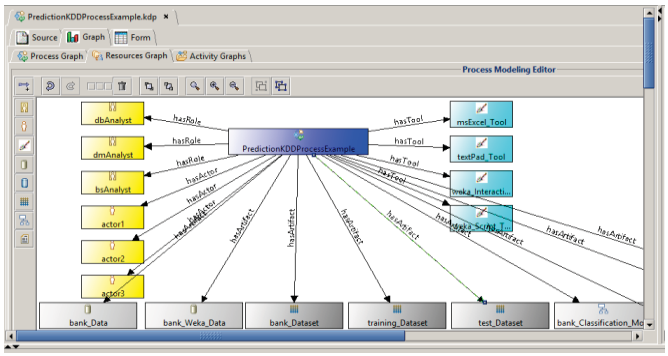


Fig. 6. The Resources Graph editor of the PUI-Modeling Perspective

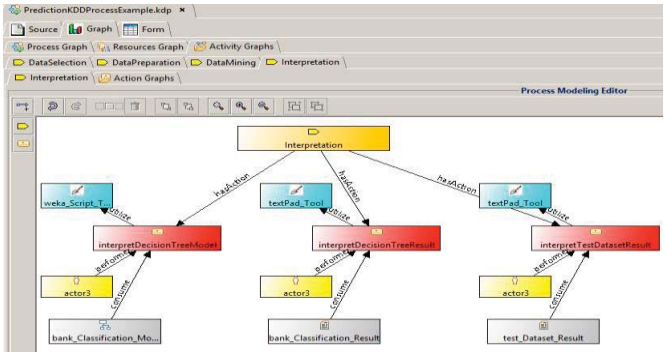


Fig. 7. The Activity Graph editor of the PUI-Modeling Perspective

2.2.2 The PUI-Repository Perspective

The PUI-Repository perspective manages the process resources using forms that are created to display and update the properties of these resources. In addition, a read-only graph is provided to show the flow of artifacts in the process starting from the process inputs to its outcomes. Fig. 8 sketches the layout of the PUI-Repository perspective.

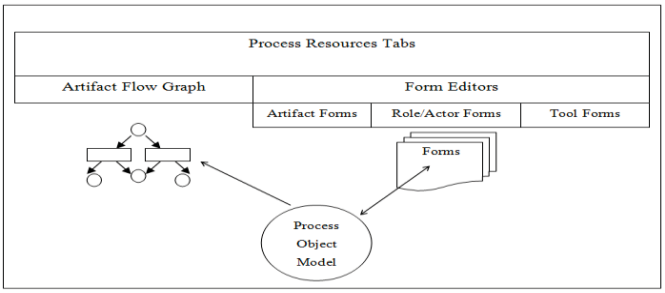


Fig. 8. The PUI-Repository Perspective

The Process Resources Tabs includes a tab for the Artifact Flow Graph and a tab for the Form editors. The Form Editors tab includes a tab for the artifact forms, a tab for the role/actor forms, and a tab for the tool forms. The Artifact Forms includes a tab for each artifact. The Role/Actor Forms includes a tab for each role/actor. The Tool Forms includes a tab for each tool. The Artifact tab includes the Artifact Form editor. The Role/Actor tab includes the Role/Actor Form editor. The Tool tab includes the Tool Form editor. Fig. 9 depicts a view of the PUI-Repository perspective showing its Artifact Flow Graph.

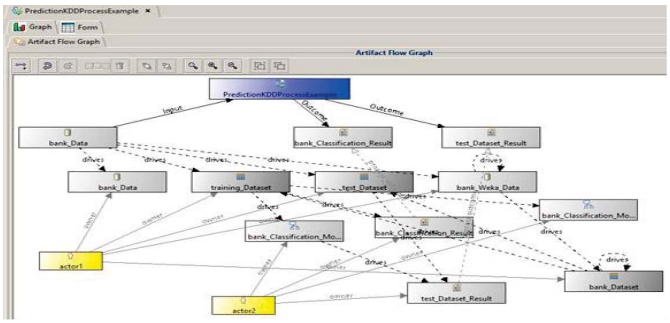


Fig. 9. The Artifact Flow Graph of the PUI-Repository Perspective

2.2.3 The PUI-Enactment Perspective

The PUI-Enactment perspective includes the items visible during the enactment phase along with their provided presentations and supported actions in the user interface. Fig. 10 sketches the layout of the PUI-Enactment perspective.

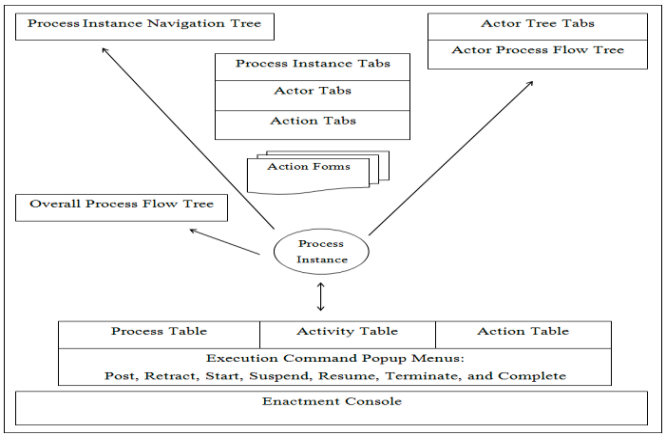


Fig. 10. The PUI-Enactment Perspective

The Process Instance Navigation Tree allows for navigating between multiple process instances for the same process or for different processes. The Overall Process Flow Tree and Actor Process Flow Tree represent process tasks as nodes that change their color based on the execution state of the task (Posted=Orange, [Started, Resumed]=Green, Suspended=Red, [Completed, Terminated]=Blue, Otherwise=Black). The Process Instance Tabs include a tab for each process instance. The Actor Tree Tabs includes a tab for each actor showing the Actor Process Flow Tree. The Actor Process Flow Tree groups and displays the tasks assigned for the actor. The Actor Tabs includes a form for each actor showing the actions assigned to the actor. The Action Forms display detailed action information. The Process Table displays process instances, the Activity Table displays activity instances, and the Action Table displays action instances. Each task in these tables is displayed with its execution state and performing actor name along with other execution information such as its running time. A popup menu is displayed showing applicable execution commands to select from. The Enactment Console displays execution information.

Fig. 11 depicts a view of the PUI-Enactment perspective.

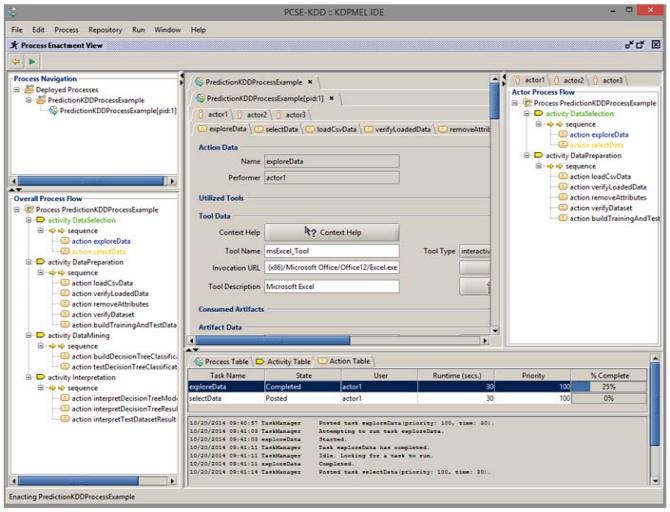


Fig. 11. Enacting a Process Instance in the PUI-Enactment Perspective

2.3 The Enactment Engine

The main components of the Enactment Engine are:

- The KDPMEL Interpreter
- The Repository Management Unit (RMU)
- The Tool Invocation Unit (TIU)

2.3.1 The KDPMEL Interpreter

The user interactions with KDPMEL Interpreter are performed through the PUI-Enactment perspective.

The execution of a process program starts by issuing an execute command, which causes the PUI-Enactment perspective to be created and presented (Fig. 11) to the user. A process instance is created, placed in a *posted* state, and entered in the process table (Fig. 12). The user is offered a menu of valid transition states; e.g., a *posted* task can be either *retracted* or *started.* The same mechanism applies for activities and actions that are ready for execution: they are placed in a *posted* state in the activity and action tables. Starting a process instance triggers the execution of its activities in the order they are defined. Starting an activity triggers the execution of its sub-activities in a depth-first order and its constituent actions based on their control construct.

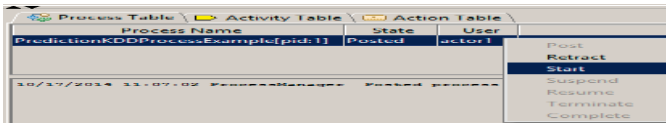


Fig. 12. A *Posted* KDPMEL Process Instance

The Interpreter interacts with the PUI-Enactment perspective to update its presentation based on the execution states and to accept the user selection of choices offered during the execution.

To illustrate the dynamics of executing a KDPMEL *action*, consider the following example for building a decision tree classification model using the WEKA [14] framework:

```
process ADecisionTreeProcess {
     resources {
          artifacts {
              Dataset sampleDataset …
              Model sampleDecisionTreeModel …
          }
          roles { Actor dmAnalyst …}
          tools { Tool weka_Script_Tool …}
     }
     …
     action buildDecisionTreeClassificationModel {
          consume sampleDataset;
          produce sampleDecisionTreeModel;
          performer dmAnalyst;
          utilize {
             call weka_Script_Tool {
              command buildDecisionTreeCommand {
                kind Modeling;
                input sampleDataset;
                output sampleDecisionTreeModel;
                operation
                     "weka.classifiers.trees.J48";
                parameters "-C 0.25 -M 2";
     …
}
```

The Interpreter establishes a handle on the action's consumed (*sampleDataset* artifact) and produced (*sampleDecisionTreeModel* artifact) artifacts by submitting queries to the RMU. Each artifact handle contains information, such as *name*, *type*, *URL*, etc., which allows accessing and controlling the artifacts. Another handle is established for each utilized tool (*weka_Script_Tool* tool). The Interpreter sends the TIU a tool handle that specifies the tool description and how to invoke it (invocation *URL*/*command*). Two mechanisms are used for tool invocations. Simple invocation is provided for an interactive tool through calling the tool's URL. A tool that can be called in a scripted mode is invoked through a plug-in module that implements the translation of KDPMEL external commands (*buildDecisionTreeCommand* command) into their appropriate commands that are accepted by the tool.

The interpreter identifies the assigned actor (*dmAnalyst* actor) by issuing a query to the RMU and accordingly notifies that actor. An action that awaits a user's response is in the *posted* state. The actor would respond through the PUI-Enactment perspective by selecting his/her preferred choice. When the actor starts the action, the interpreter identifies the needed artifacts, binds them with the action, identifies the tools utilized by the action, and issues their invocation calls. The actor then takes responsibility for executing and finishing the action and informing the interpreter through the PUI-Enactment perspective about the completion status. If the action is completed successfully, the interpreter issues an update or create request to the RMU for the produced artifacts and determines the next action to be executed.

When *buildDecisionTreeClassificationModel* action becomes ready for execution, it is placed in a *posted* state on the action table. The color of the node representing the action in the Overall Process Flow Tree and Actor Process Flow Tree changes to *Orange*. The actor performing the action starts its execution by clicking on the row representing the action in the action table and selecting the *Start* choice from the popup menu. A number of dialogs begin with the actor to support the performance of the action, the state of the action changes to *started* state, and the action node color changes to *Green*. The first dialog asks for invoking the *weka_Script_Tool* tool that is utilized by the action and the actor responds with *Yes*. The second dialog asks whether to run the commands associated with the tool and the actor responds with *Yes*. The third dialog asks to run the first tool command (*buildDecisionTreeCommand* command) and the actor responds with *Yes*. The last dialog confirms that the command has been executed successfully. Fig. 13 illustrates the execution of the action.

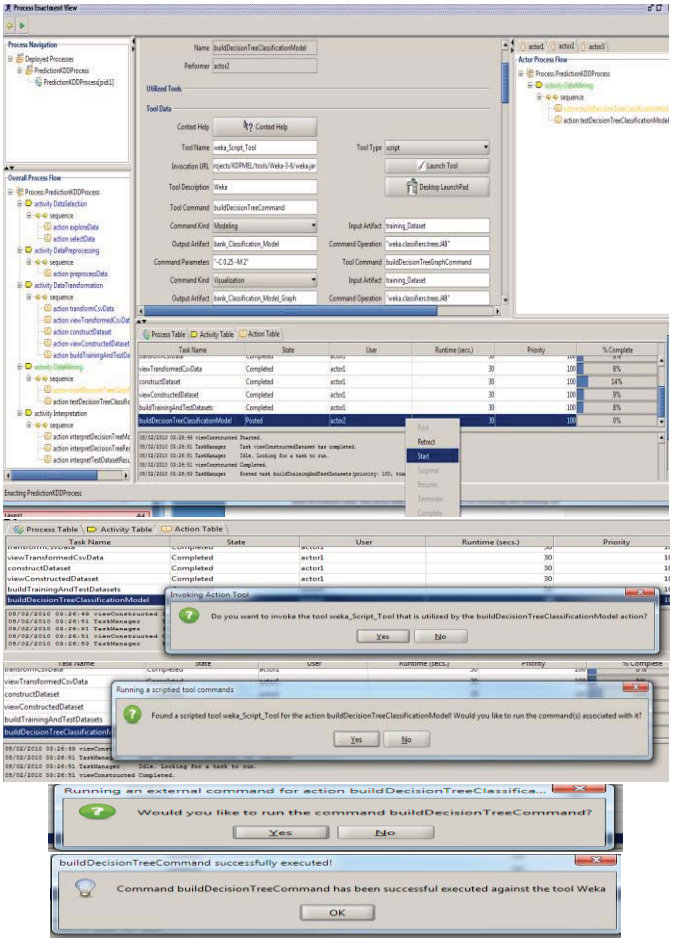


Fig. 13. The execution of the *buildDecisionTreeClassificationModel* action

2.3.2 The Repository Management Unit (RMU)

RMU provides management and persistence support for process resources. In addition, process instances that are created during enactment are maintained by the RMU.

Concurrent access to the process artifacts can happen in a number of situations and there is a need for a concurrency control mechanism, suitable for KDD processes, to handle these situations. KDD processes are highly interactive and iterative. They include tasks that can have very long running times and a high degree of interactions with process actors.

These KDD tasks are similar to database transactions that involve browsing or performing data entry, which may last several minutes. For this kind of transactions, concurrency can severely suffer with higher isolation levels. Also, the possibility of deadlock is increased. A lower isolation level is typically used for this kind of transactions.

Given the nature of KDD tasks and their running time similarities with long running database transactions, the lowest isolation level would be a better choice. The RMU supports the lowest isolation level as defined in the ANSI SQL Standard [6] by implementing a simple exclusive locking mechanism on the process artifacts that are updated by an action. Process artifacts that are updated by an action are locked for exclusive use (write-lock) when the action is started (the *started* state) and released when the action is finished with either *terminated* or *completed* state.

2.3.3 The Tool Invocation Unit (TIU)

TIU is responsible for orchestrating and managing the invocation of tools. Two mechanisms are used for tool invocations. Simple invocation is provided for an interactive tool through calling the tool's URL. A tool that can be called in a scripted mode is invoked through a plug-in module that implements the translation of KDPMEL external commands into their appropriate commands that are accepted by the tool. Fig. 14 illustrates the structure of the TIU.

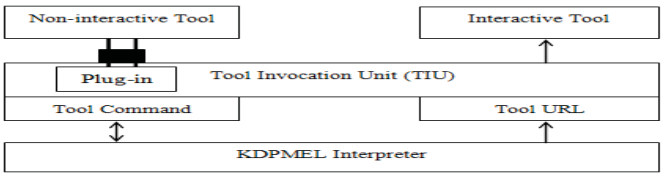


Fig. 14. The Tool Invocation Unit (TIU)

2.4 PCSE-KDD Key Aspects and Novelty

PCSE-KDD combines aspects form KDD Support Environments, Language Integrated Development Environments (IDEs), and PCSEEs. These aspects are blended together in PCSE-KDD to support developing KDD processes in a language-based and process-oriented approach.

As a KDD support environment, PCSE-KDD provides features to assist KDD developers, maintain KDD resources, invoke and interact with KDD supporting tools, automate routine KDD tasks, manage dependencies and interactions between KDD techniques, and enforce mandatory KDD rules and practices.

As an IDE for KDPMEL, PCSE-KDD provides basic features to facilitate the construction, execution, and management of KDPMEL programs. KDPMEL programs are constructed using a hybrid modeling approach, mixing different types of editors and views in source-based, graph-based, and form-based styles.

As a PCSEE, PCSE-KDD exploits an explicit representation of the KDD process (a process program) to support KDD development activities and to manage the overall process. PCSE-KDD implements the process definition/instantiation/enactment paradigm to develop KDD processes in a way similar to developing software processes in PCSEEs.

We believe that this novel approach for designing PCSE-KDD combines the benefits of the underlying aspects (KDD, Language, and Process) to provide concrete, flexible, explicit, and process-oriented support for KDD processes.

We have used PCSE-KDD to implement a non-trivial KDD prediction process in [1]. To evaluate PCSE-KDD, we have also implemented a real-world case study data analysis process for analyzing, comparing, and visualizing streams of ocean data [3]. The evaluation results of this case study show that the entire process can be fully automated, which yields an execution time of 4.8 hours as opposed to 79 hours original execution time for the manual non-process implementation. This is a 16.5x speedup over the original execution time. The results also show that the execution of the process can be easily repeated with the same or different configurations and/or data. Moreover, the results show that minor to moderate program adjustments and configurations are needed to expand and scale up the analysis. As for novice users, the results show that they will have no difficulty in executing the analysis in PCSE-KDD.

3 Prototyping PCSE-KDD

The environment has been prototyped in Java utilizing a number of open source libraries and tools such as *JavaCC* [5], *JGraph* [27], *JGoodies* [16], and *SMC* [15]. The environment has the look and feel of Eclipse IDE. It also has similar Workbench that includes three different perspectives for the Modeling, Enactment, and Management functionalities.

The prototype is structured into multiple modules: the KDPMEL language along with its components and tools (Parser, Model Checker, Interpreter, Source-code Editor, etc.); the Process Object Model (Process Components); the Desktop Development Environment (Workbench, Graph Editors, and Form Editors); the Runtime System (Enactment Engine and State Diagrams); and the Repository.

In addition to the PCSE-KDD Java prototype, an Eclipse Rich Client Platform (RCP) Plug-in version has been prototyped to provide full modeling and management capabilities that are supported by the Eclipse platform. With Eclipse RCP, a plethora of Eclipse features and components are available for reuse. Building on a platform facilitates faster development and seamless integration. The inherit extensibility of Eclipse allows to build not only a closed-form application, but also and open-ended platform like the Eclipse IDE itself. This RCP version utilizes the following technologies:

- The Eclipse Workbench, Perspectives, Views, and Editors for building the Desktop of the environment.

- The Eclipse Modeling Framework (EMF) [11] for defining the KDD Process Meta-Models.
- The Eclipse Graphical Modeling Framework (GMF) [12] and Graphical Editing Framework (GEF) [13] for building the KDD Process Graph Editors.
- The *xText* Language Development Framework [17] for developing KDPMEL.

4 Conclusions

In this paper, we presented the Process-Centered Support Environment for KDD (PCSE-KDD) that can be used to develop KDD processes in a way that is similar to developing software processes, which is based on encoding KDD processes as process programs written in KDPMEL and exploited by PCSE-KDD to provide execution support and management for KDD processes.

PCSE-KDD includes a number of modeling editors and views, an Enactment Engine for runtime process execution support, and a Repository for providing persistence support for the process resources. PCSE-KDD has been prototyped in Java plus a number of open source libraries and tools.

In PCSE-KDD, the process concept is supported and enforced according to a specialized KDD process that includes specific tasks organized according to their sequencing, dependencies, and alternatives. Also, tools are loosely integrated through a flexible and expandable plug-in mechanism. They are launched automatically and dynamically according to the execution order of the process tasks. PCSE-KDD employs an engineering approach to develop KDD processes. It is a language-based and process-driven approach. In this language-based approach, KDD processes are managed. Their specifications can evolve and executions can be repeated. Moreover, they are validated according to standard programming techniques.

Our future work includes expanding the support for more KDD tools and continuing the development of PCSE-KDD to provide more enhanced graphical modeling and management for KDD artifacts.

5 References

[1] Mansour, H. A., Duchamp, D., and Krapp, C.-A. *A Language-Based and Process-Oriented Approach for Supporting the Knowledge Discovery Processes*. In Proceedings of the 11th International Conference on Data Mining (DMIN'15) (pp. 107-115), July 2015.

[2] Mansour, H. A. (in press). *KDPMEL*: *A Knowledge Discovery Process Modeling and Enacting Language*. The 12th International Conference on Data Mining (DMIN'16), July 2016.

[3] Mansour, H. A. *A Process-Centered Environment for Modeling, Enacting, and Managing the Knowledge Discovery Processes*. PhD dissertation, Stevens Institute of Technology, Hoboken, N. J., 2015.

[4] David Jensen et al. *Coordinating Agent Activities in Knowledge Discovery Processes*, Department of Computer Science, University of Massachusetts Amherst, 1999.

[5] The *JavaCC* Framework. URL: https://javacc.dev.java.net/

[6] The SQL-92 Standard. URL: http://www.contrib.andrew.cmu.edu/~shadow/sql/sql1992.txt.

[7] L. Kurgan and P. Musilek. *A Survey of Knowledge Discovery and Data Mining Process Models*. Knowledge Engineering Review, 21(1), pp. 1-24, 2006.

[8] Marban, O. et al. *An Engineering Approach to Data Mining Projects. Intelligent Data Engineering and Automated Learning* – IDEAL 2007, LNCS 4881, pp. 578-588, 2007.

[9] Marban, O. et al. *Toward data mining engineering: A software engineering approach*. Information Systems 34 (1), 2009.

[10] Pohl, K. et al. *PRIME-Toward Process-Integrated Modeling Environments*. ACM Transactions on Software Engineering and Methodology, Vol. 8, No. 4, October 1999, Pages 343-410.

[11] The Eclipse Modeling Framework (EMF). URL: http://www.eclipse.org/modeling/emf/

[12] The Eclipse Graphical Modeling Framework (GMF). URL: http://wiki.eclipse.org/Graphical_Modeling_Framework/

[13] The Graphical Editing Framework (GEF). URL: http://www.eclipse.org/gef/

[14] University of Waikato, New Zealand. Weka 3: Data Mining Software in Java. URL: http://www.cs.waikato.ac.nz/ml/weka/

[15] Open Source, The State Machine Compiler (SMC) Framework, URL: http://smc.sourceforge.net/

[16] The JGoodies Framework. URL: https://jgoodies.dev.java.net/

[17] The *xText* Language Development Framework. URL: http://www.eclipse.org/Xtext/

[18] Rudiger Wirth et al. *Towards Process-Oriented Tool Support for Knowledge Discovery in Databases*. DaimlerChrysler Research & Technology, 1997.

[19] Cinara Ghedini and Karin Becker. *A documentation model for the KDD application management support*. Faculdade de Informatica, PUCRS 2000.

[20] Osterweil, L.J. *Software Processes are Software Too*. In Proceedings of the Ninth International Conference on Software Engineering, pp 2-14, 1987.

[21] B.C. Warboys et al. *Collaboration and Composition: Issues for a Second Generation Process Language*. 1999.

[22] Vincenzo Ambriola et al. *Assessing Process-centered Software Engineering Environments*. Universita di Pisa, NTH-Trondheim, Politecnico di Milano, 1996.

[23] Padhraic Smyth. *Breaking Out of the Black-Box: Research Challenges in Data Mining*. The 2001 ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, 2001.

[24] Fayyad U. M., Piatetsky-Shapiro, G., and Uthurusamy, R. *Summary from the KDD-03 Panel – Data Mining: The Next 10 Years*. The 9th International Conference on Data Mining and Knowledge Discovery: KDD-03, August 27, 2003.

[25] J. Segovia. *Definition and Instantiation of an Integrated Data Mining Process*. Jornadas de Seguimiento de Proyectos, 2007.

[26] Using Perspectives in the Eclipse UI. URL: https://www.eclipse.org/articles/using-perspectives/PerspectiveArticle.html

[27] The JGraph Framework. URL: http://www.jgraph.com/jgraph.html

Robust Speaker Recognition in the Presence of Speech Coding Distortion for Remote Access Applications

Robert W. Mudrowsky[1], Ravi P. Ramachandran[1], Umashanger Thayasivam[2] and Sachin S. Shetty[3]
[1] Department of Electrical and Computer Engineering, Rowan University, Glassboro, NJ, USA
[2] Department of Mathematics, Rowan University, Glassboro, USA
[3] Department of Electrical and Computer Engineering, Tennessee State University, Nashville, TN, USA

***Abstract*— For wireless remote access security, forensics, border control and surveillance applications, there is an emerging need for biometric speaker recognition systems to be robust to speech coding distortion. This paper examines the robustness issue for three codecs, namely, the ITU-T 6.3 kilobits per second (kb/s) G.723.1, the ITU-T 8 kb/s G.729 and the 12.2 kb/s 3GPP GSM-AMR coder. Both speaker identification and speaker verification systems are considered and use the Gaussian mixture model classifier. The systems are trained on clean speech and tested on the decoded speech. To mitigate the performance loss due to mismatched training and testing conditions, four robust features, two enhancement approaches and fusion strategies are used. A detailed two-way ANOVA analysis is performed.**

Keywords: speaker recognition, robust system, fusion, ANOVA

I. Introduction and Motivation

Robust speech processing for wireless applications is an active area of research especially in terms of gaining remote access to voice activated services [1] (like performing a bank account transaction). This includes remote access using mobile telephony and the internet. Although most practical applications are aimed at a wireless internet connection, the technical approach described in this paper can be applied to a wired connection. In this context, the particular applications of speaker recognition [1][2][3][4][5][6][7][8] and speech recognition have been investigated [9][10]. The challenge is the diminished performance due to background noise, speech coding distortion and communication channel errors. In this paper, biometric speaker recognition in the presence of speech coding distortion is addressed. Both speaker identification (SI) (determining the most likely speaker among a set of candidates) and speaker verification (SV) (accepts or rejects a claimed identity) systems are described. The systems are both text-independent and language-independent.

The operation of the proposed system is as follows. A person seeking remote access speaks into the microphone of his/her wired node (desktop PC) or mobile/wireless node (laptop or smart phone). This speech signal is communicated to a remote server via the internet and is subject to speech coding distortion. The remote server uses a biometric based speaker recognition algorithm. The algorithm can either accomplish (1) speaker identification (SI) in which a person is identified among a set of candidates or (2) speaker verification (SV) that accepts or rejects the claimed identity of a person. The database acts as the central storage for all biometric data. The communication between the database and the server is secure and encrypted.

The speech coders [11] investigated in this paper include the ITU-T 6.3 kilobits per second (kb/s) G.723.1 coder [12][13], the ITU-T 8 kb/s G.729 coder [14] and the 12.2 kb/s 3GPP GSM-AMR coder [15]. The three coders are based on the Code Excited Linear Prediction (CELP) format. The G.723.1 and the G.729 are used in voice over internet protocol (VoIP) applications. The GSM-AMR is widely used in cellular telephony. The ITU implementations are used for the G.729 and GSM-AMR coders. The implementation in [13] is used for the G.723.1 coder.

Training of the SI and SV systems is done on clean speech. Testing (performance evaluation) is done on the decoded speech which is the clean speech passed through the speech coder and then, decoded. The systems are based on a Gaussian mixture model (GMM) classifier. A speaker independent universal background model (UBM) is first configured [8][16] using the expectation maximization (EM) algorithm [18][19]. Individual speaker models are obtained by maximum a posteriori (MAP) estimation of the UBM parameters using the training speech of the speaker [17]. Four robust features, namely, the linear predictive cepstrum (CEP), the adaptive component weighted (ACW) cepstrum [20], the postfilter (PFL) cepstrum [20] and the mel frequency cepstrum (MFCC) [3][4] are compared.

Since the mismatch between training and testing leads to diminished performance, feature enhancement using the affine transform and signal enhancement using the McCree technique [8] are individually investigated and combined. Fusion of the four features [21][22] is also attempted. Statistical methods based on analysis of variance (ANOVA) [23] will compare the performance of the individual features, fusion methods and feature and signal enhancement. The aim is to examine the relative performance of the various approaches as a function of the bit rate. This paper extends the work in [6] by examining three speech coders, implementing both SI and SV and performing two-way ANOVA.

II. System Description and Experimental Protocol

The TIMIT database is used for the experiments. The speech in this database is clean and first downsampled from 16 kHz to 8 kHz. The UBM is first configured using all ten sentences from each of 168 speakers. The k-means algorithm is used to initialize the parameters of the UBM with a diagonal covariance matrix assumed. Ten iterations of the EM algorithm results in the final UBM. Ninety individual

GMM speaker models (different speakers from those used to train the UBM) are obtained by MAP adaptation of the UBM parameters. Both adaptation of the (1) weights, mean vectors and elements of the covariance matrix and (2) only the mean vectors were attempted but no significant difference in performance was obtained. As in [17], only adaptation of the mean vectors is performed. Eight of the ten sentences for each speaker are used to obtain the feature vectors and perform the MAP estimation. Since four different features are investigated, there are four different speaker recognition systems.

For testing the system, the two sentences from each of the 90 speakers not used in training are processed. These sentences are the decoded speech from a particular speech coder. The overall system consists of five components, namely, (1) Signal enhancement using the McCree method, (2) Feature extraction for ensuring speaker discrimination, (3) Feature compensation using the affine transform that maps the test feature vectors to the space reflecting the training condition, (4) GMM classifier and decision logic and (5) Feature fusion.

A. McCree Technique

The basic steps as outlined in [8] were implemented, namely, (1) Perform linear predictive (LP) analysis of the decoded speech to obtain the nonrecursive $A(z)$, (2) Pass the decoded speech through $A(z)$ to remove the distorted spectral envelope and (3) Perform LP synthesis filtering with the transmitted LP information of the input speech to the coder in order to restore the correct spectral envelope.

B. Feature Extraction

The speech is preemphasized by using a nonrecursive filter $1-0.95z^{-1}$ and then divided into frames of 30 ms duration with a 20 ms overlap. A 12th order LP analysis is performed using the autocorrelation method. The LP coefficients are converted into 12 dimensional CEP, ACW and PFL feature vectors. The PFL feature is obtained by weighting the CEP feature by the factor $[1-0.9^n]$ for $n=1$ to 12 [20].

A 12 dimensional MFCC feature vector is computed in each frame. The success of the MFCC is due to the perceptually based filter bank processing of the Fourier transform of the speech followed by cepstral analysis using the DCT [3][4].

For each of the four features, a 12 dimensional delta feature [3] is computed in each frame using a frame span of 5 in order to derive first derivative information. Second derivative information is also obtained. Concatenation of the feature vector, the first derivative and the second derivative results in a 36 dimensional vector that is used for speaker recognition.

Energy thresholding is performed over all frames of an utterance to determine the relatively high energy speech frames. This is the most simple form of voice activity detection [5]. The 36 dimensional vectors are considered only in these high energy frames.

C. Affine Transform

The affine transform accomplishes feature compensation by mapping a feature vector $\mathbf{x}$ derived from test speech to a feature vector $\mathbf{y}$ in the region of the p-dimensional space occupied by the training vectors. This forces a better match between training and testing conditions and in effect, compensates for the distortion of the test speech (in this case, the codec). The transform relating $\mathbf{x}$ and $\mathbf{y}$ is given by Eq. (1) as

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b} \tag{1}$$

where $\mathbf{A}$ is a p by p matrix and $\mathbf{y}$, $\mathbf{x}$ and $\mathbf{b}$ are column vectors of dimension p. Expanding Eq. (1) gives

$$\begin{bmatrix} y(1) \\ y(2) \\ y(3) \\ \vdots \\ y(p) \end{bmatrix} = \begin{bmatrix} \mathbf{a}_1^T \\ \mathbf{a}_2^T \\ \mathbf{a}_3^T \\ \vdots \\ \mathbf{a}_p^T \end{bmatrix} \begin{bmatrix} x(1) \\ x(2) \\ x(3) \\ \vdots \\ x(p) \end{bmatrix} + \begin{bmatrix} b(1) \\ b(2) \\ b(3) \\ \vdots \\ b(p) \end{bmatrix} \tag{2}$$

where $\mathbf{a}_m^T$ is the row vector corresponding to the mth row of $\mathbf{A}$.

The affine transform parameters $\mathbf{A}$ and $\mathbf{b}$ are determined from the five (the "sa" and "si") sentences of each of the 90 speakers under consideration. The affine transform is computed only during training and is different for each feature and for each of the three codecs. A total of 12 affine transforms are computed. It is only computed for the 12 dimensional feature vector and not for the first and second derivative information.

Let $\mathbf{y}^{(i)}$ be the feature vector for the ith frame of the training speech utterance. Let $\mathbf{x}^{(i)}$ be the feature vector for the ith frame of the training speech utterance passed through the distortion encountered during testing (in this case, the clean utterance is compressed and decoded by the codec). By using a number of training speech utterances, N sets of vectors are collected, namely, $\mathbf{y}^{(i)}$ and $\mathbf{x}^{(i)}$ for $i=1$ to N. A squared error function is formulated as

$$E(m) = \sum_{i=1}^{N} \left[y^{(i)}(m) - \mathbf{a}_m^T \mathbf{x}^{(i)} - b(m) \right]^2 \tag{3}$$

where $\mathbf{a}_m^T$ is the mth row of $\mathbf{A}$ and $y^{(i)}(m)$ and $b(m)$ are the mth components of $\mathbf{y}^{(i)}$ and $\mathbf{b}$, respectively. Minimization of $E(m)$ with respect to $\mathbf{a}_m^T$ and $b(m)$ results in the system of equations [6]

$$\begin{bmatrix} \sum_{i=1}^{N} \mathbf{x}^{(i)} \mathbf{x}^{(i)T} & \sum_{i=1}^{N} \mathbf{x}^{(i)} \\ \sum_{i=1}^{N} \mathbf{x}^{(i)T} & N \end{bmatrix} \begin{bmatrix} \mathbf{a}_m \\ b(m) \end{bmatrix}$$

$$= \begin{bmatrix} \sum_{i=1}^{N} y^{(i)}(m)\mathbf{x}^{(i)} \\ \sum_{i=1}^{N} y^{(i)}(m) \end{bmatrix}. \quad (4)$$

The function $E(m)$ is minimized for $m = 1$ to p. Hence, m different systems of equations of dimension $(p + 1)$ as described by Eq. (4) are solved. The left hand square matrix of Eq. (4) is independent of m and hence, needs to be computed only once.

D. GMM Classifier and Decision Logic

A GMM speaker model λ is completely specified by a conditional probability density expressed as a linear combination of Gaussian densities as given by

$$p(\mathbf{x}|\lambda) = \sum_{i=1}^{V} w_i c_i(\mathbf{x}) \quad (5)$$

where $\mathbf{x}$ is a p-dimensional feature vector, $c_i(\mathbf{x})$ is a p-variate Gaussian probability density function and w_i are the mixture weights ($\sum w_i = 1$) for $i = 1$ to V (V is the number of Gaussian mixtures). The Gaussian density $c_i(\mathbf{x})$ is specified by a vector of means μ_i and a covariance matrix Σ_i.

A test speech utterance is processed such that a set of test feature vectors $X = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_q)$ are computed. In computing these feature vectors, the option of using or not using the McCree technique and the affine transform exists. For the SI system, there are $M = 90$ speakers for which speaker i is represented by GMM λ_i, the identified speaker $M^\star$ is chosen to maximize the a posteriori log-probability as given by

$$M^\star = \arg \max_{1 \le j \le M} \sum_{i=1}^{q} \log p(\mathbf{x}_i|\lambda_j) = \arg \max_{1 \le j \le M} d(j) \quad (6)$$

where $p(\mathbf{x}_i|\lambda)$ is computed as given in Eq. (5) and $d(j)$ is the SI score for speaker j. When many utterances are tested, the identification success rate (ISR) is the number of utterances for which the speaker is identified correctly divided by the total number of utterances tested. For each codec, a total of 180 utterances are tested to get the ISR.

For the test utterance for the SV system, let the claimed identity be speaker k. The a posteriori log-probability, as in Eq. (6) is calculated for the speaker model λ_k and for the UBM model. These two quantities are subtracted to yield the SV score. The SV score is compared to a threshold to either accept of reject the claimed identity. Different thresholds are attempted to yield a receiver operating characteristic (ROC) from which the equal error rate (EER) is the performance measure. For each codec, there will be 180 genuine attempts and $(180)(89) = 16,020$ impostor attempts.

For each speaker, the 2 "sa" and three "si" sentences are always used in training the GMM speaker model as they are used in computing the affine transform. Of the remaining five "sx" sentences, three are rotated for training purposes and the other two are used for testing. This enables 10 trials for each experimental condition.

E. Feature Fusion

Since four different features are used, each with a separate GMM classifier, an ensemble system [21] results which naturally leads to the investigation of fusion. For SI systems, decision level fusion is the simplest technique and involves taking a majority vote of the four different features to get a final decision. The second fusion method is to use Borda count based on the SI scores $d(i)$ for $i = 1$ to $M = 90$ (see Eq. (6)) obtained for the four features. For the SV system, the genuine and impostor SV scores are normalized to be in the range [0,1] for each feature separately. Fusion of these normalized SV scores for the four features are used to obtain the EER. The three methods of fusion considered are the sum of the scores, product of the scores and taking only the maximum score.

III. General Results

The first aspect was to examine how many mixtures to use for the UBM and the individual speaker models. The performance of the four features for all conditions (clean, codec only and using the McCree method and/or the affine transform) was compared when the number of mixtures was varied from 16 to 2048, each time doubling the number of mixtures. The use of 256 and 512 mixtures yielded the best and statistically comparable performance. Using more than 512 mixtures did not necessarily result in higher performance. The results continue for the case of using 256 mixtures.

Tables I and II show the speaker identification and verification results respectively. The average performance over 10 trials is given.

IV. Two-way ANOVA

The balanced two-way ANOVA model (two factors) is considered. A model is considered for each codec separately and within each codec, the SI and SV scenarios separately. There is a continuous response, Y, and two factors, A and B. In this case, Y refers to either the ISR or EER. Factor A are the methods used and has $a = 4$ levels, namely, coder distortion only, coder distortion mitigated by the McCree method, coder distortion mitigated by the affine transform and coder distortion mitigated by both. Factor B are the features and the fusion methods and has b levels. With two fusion methods for SI, $b = 6$. With three fusion methods for SV, $b = 7$. The interests lie in determining whether or not the response (performance) differs due to the two factors and the interaction between the factors. For the SI and SV systems involving each codec, ANOVA revealed that there is a significant interaction between the methods and the features.

Experiments with multiple factors will generally look at a factorial treatment structure. This implies 'treatments' are combinations of the levels of different factors. The experiments in this study are **complete** in that there are

Condition	CEP	ACW	PFL	MFCC	Decision Level Fusion	Borda Count
Clean	93.1	93.2	92.1	95.4	95.2	95.2
G.723.1	64.6	62.5	65.3	79.3	69.0	72.3
G.723.1, McCree	70.4	67.6	71.2	83.0	75.3	77.3
G.723.1, Affine	77.7	74.2	77.7	86.3	83.4	84.3
G.723.1, McCree and Affine	82.8	78.9	80.7	85.8	86.5	87.9
G.729	65.7	61.4	64.6	78.5	69.9	70.2
G.729, McCree	85.0	83.5	83.6	91.1	88.3	89.3
G.729, Affine	84.3	80.9	82.1	89.3	87.8	88.9
G.729, McCree and Affine	86.8	85.5	86.7	90.3	90.2	91.1
GSM-AMR	75.9	73.8	75.3	78.9	78.9	76.3
GSM-AMR, McCree	86.1	83.7	84.2	84.2	87.7	84.4
GSM-AMR, Affine	86.5	86.6	86.2	84.0	89.8	85.3
GSM-AMR, McCree and Affine	85.3	84.2	83.8	83.6	88.2	83.8

TABLE I
IDENTIFICATION SUCCESS RATE (%) FOR VARIOUS CONDITIONS (AVERAGE OVER 10 TRIALS)

Condition	CEP	ACW	PFL	MFCC	Sum Fusion	Product Fusion	Maximum Score Fusion
Clean	3.61	3.35	3.39	3.13	2.78	2.77	3.40
G.723.1	8.43	8.79	8.87	5.98	6.48	6.65	6.62
G.723.1, McCree	7.87	8.09	7.67	5.43	5.75	5.88	6.01
G.723.1, Affine	5.75	6.61	6.22	4.59	4.60	4.56	5.27
G.723.1, McCree and Affine	4.95	5.73	5.51	4.29	4.10	4.13	4.74
G.729	8.11	8.59	8.44	6.69	6.44	6.57	6.63
G.729, McCree	4.82	4.85	4.80	3.90	3.74	3.67	4.12
G.729, Affine	5.29	4.85	4.79	4.07	3.79	3.77	4.38
G.729, McCree and Affine	4.05	4.04	3.93	3.51	3.13	3.19	3.77
GSM-AMR	6.63	6.18	6.25	4.61	4.90	4.90	5.77
GSM-AMR, McCree	5.38	4.86	4.94	3.29	3.51	3.47	4.55
GSM-AMR, Affine	4.65	4.58	4.55	3.44	3.26	3.24	4.37
GSM-AMR, McCree and Affine	5.34	4.96	4.96	3.39	3.66	3.58	4.89

TABLE II
EQUAL ERROR RATE (%) FOR VARIOUS CONDITIONS (AVERAGE OVER 10 TRIALS)

observations at each level combination. The parameterization of the Two-way ANOVA model is

$$Y_{ijk} = \mu + \alpha_i + \beta_j + (\alpha\beta)_{ij} + E_{ijk} \quad (7)$$

for $i = 1, ..., a$, $j = 1, ..., b$, and $k = 1, ..., n_{ij}$. Normality is assumed for the errors. The parameter constraints sum to zero in order to get a unique solution. The variable Y_{ijk} is the k^{th} observation on the ij^{th} treatment, where $k = 1, ..., n_{ij}$ and n_{ij} is the size of the sample drawn from each treatment (this is a balanced design so all treatment samples are of size n). The variable μ denotes the global mean. The quantity α_i is the differential effect of the ith level of factor A on the mean response (the mean of the dependent variable) such that $\sum_i(\alpha_i) = 0$. The quantity β_j is the differential effect of the jth level of factor B on the mean response (the mean of the dependent variable) such that $\sum_j(\beta_j) = 0$. The quantity $(\alpha\beta)_{ij}$ is the differential effect of the ijth treatment on the mean response such that $\sum_i(\alpha\beta)_{ij} = 0$ and $\sum_j(\alpha\beta)_{ij} = 0$. The error is denoted by E_{ijk}.

A. Speaker Identification

A discussion of the comparison among the methods and features individually is given. Considering the interaction between the methods and features, the best approaches are also mentioned.

1) G.723.1: Figure 1 and 2 show the 95% confidence interval for the methods and features respectively. It is clear that combining the McCree technique and the affine transform is the best method. The features (includes decision level fusion and Borda count) are similarly compared and the best feature is the MFCC. Due to the interaction of the feature and method, the best performance (average ISR of 87.9%) is obtained using the McCree technique and the affine transform in conjunction with Borda count fusion.

2) G.729: Figures 3 and 4 show the 95% for the methods and features respectively. As in the case of G.723.1, (1) the best method is to combine the McCree technique and the

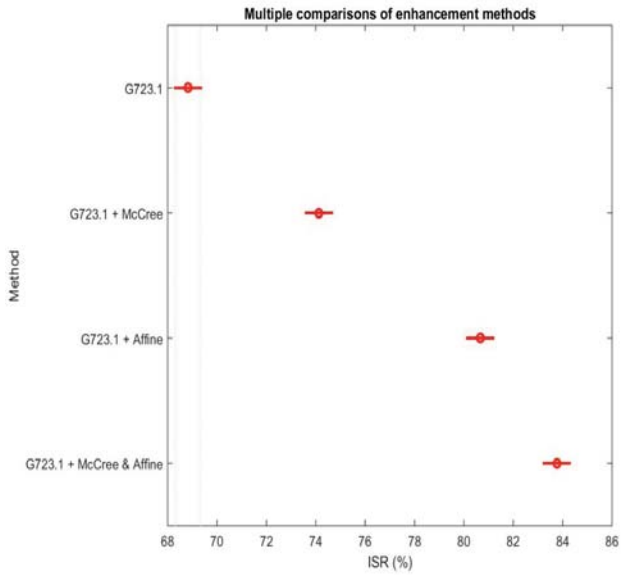


Fig. 1. Speaker identification: Comparison of the methods (G.723.1).

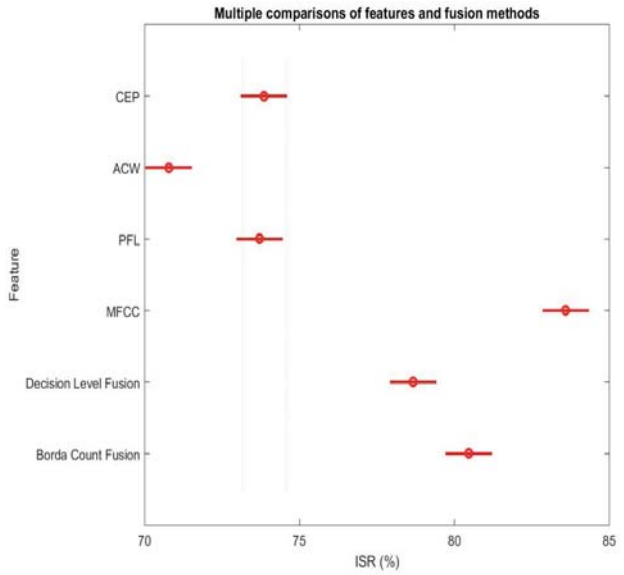


Fig. 2. Speaker identification: Comparison of the features (G.723.1).

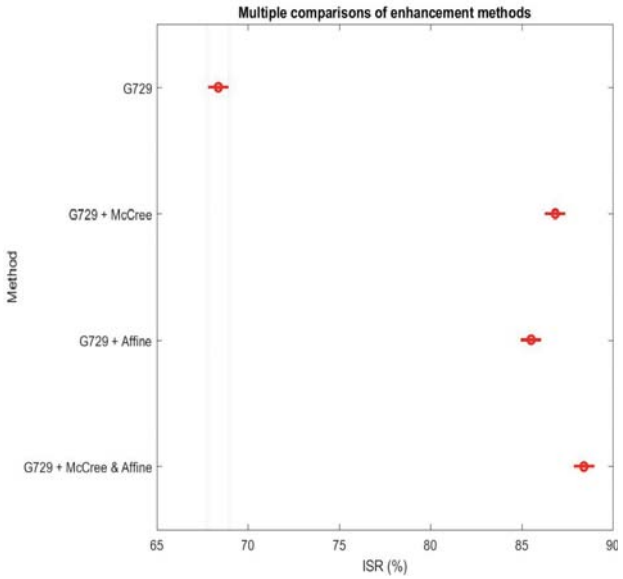


Fig. 3. Speaker identification: Comparison of the methods (G.729).

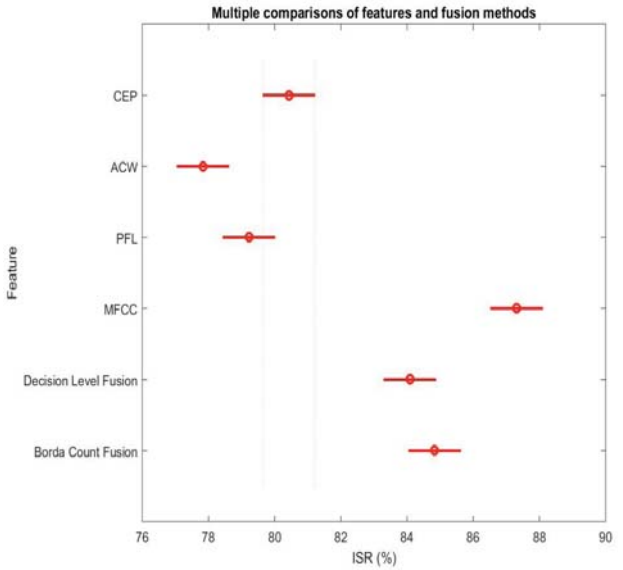


Fig. 4. Speaker identification: Comparison of the features (G.729).

affine transform and (2) the best feature is the MFCC. Due to the interaction of the feature and method, the best performance (average ISR of 91.1%) is obtained using either (1) the McCree technique and the affine transform in conjunction with Borda count fusion or (2) the McCree technique with the MFCC feature.

3) GSM-AMR: Figure 5 and 6 show the results. The best method is using only the affine transform. The best feature is the use of decision level fusion. It is the same two approaches that interact the best achieving an average ISR of 89.8%.

B. Speaker Verification

Results based on the EER are given for the SV system.

1) G.723.1: Figure 7 and 8 show the 95% confidence interval for the methods and features respectively. It is clear that combining the McCree technique and the affine transform is the best method. The features (includes sum, product and maximum score fusion) are similarly compared. Although the best feature is the MFCC, its 95% confidence interval overlaps with that of sum and product fusion. Due to the interaction of the feature and method, the best performance (average EER of 4.1%) is obtained using the McCree technique and the affine transform in conjunction with sum fusion. Using product fusion is statistically comparable and leads to only a slightly higher average EER of 4.13%.

2) G.729: Figures 9 and 10 show the 95% confidence interval for the methods and features respectively. The best method is to combine the McCree technique and the affine transform. Although the sum fusion is the best feature, its

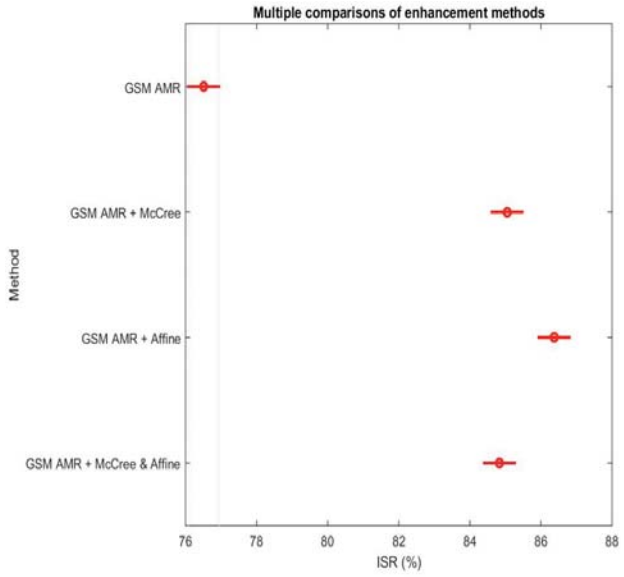


Fig. 5. Speaker identification: Comparison of the methods (GSM-AMR).

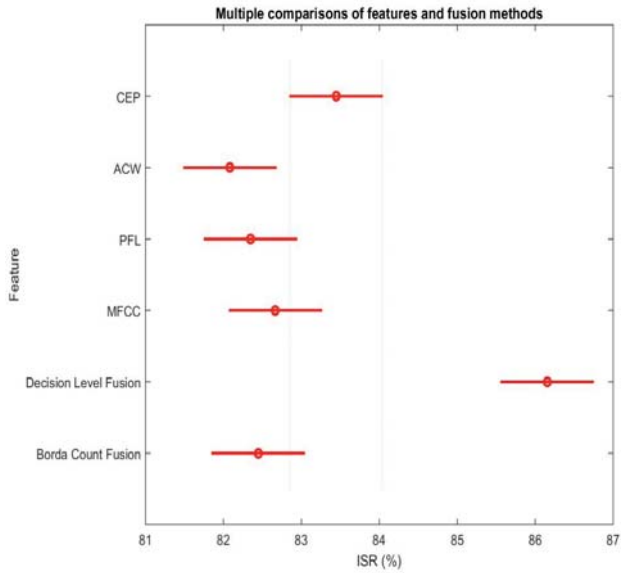


Fig. 6. Speaker identification: Comparison of the features (GSM-AMR).

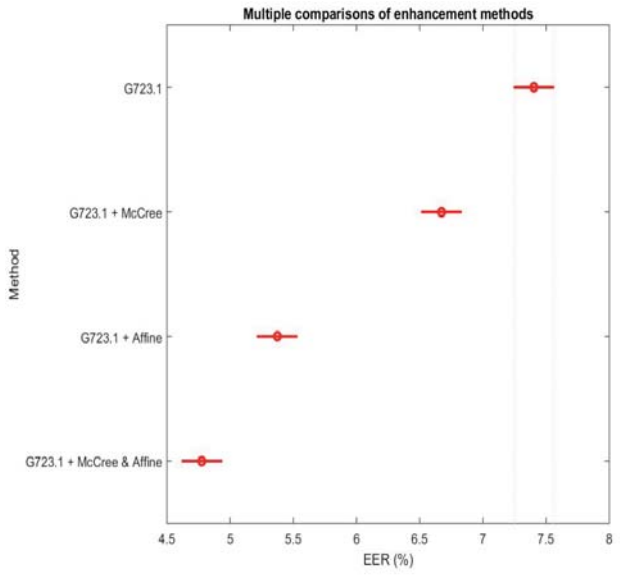


Fig. 7. Speaker verification: Comparison of the methods (G.723.1).

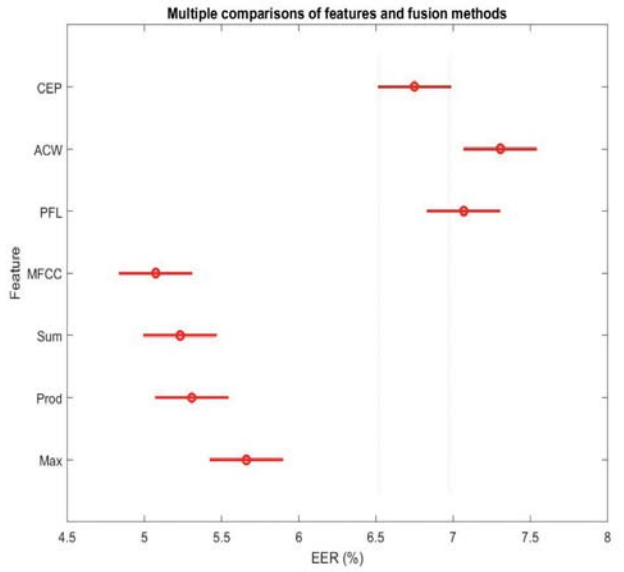


Fig. 8. Speaker verification: Comparison of the features (G.723.1).

95% confidence interval has considerable overlap with the product fusion and partial overlap with the MFCC. Due to the interaction of the feature and method, the best performance (average EER of 3.13%) is obtained using the McCree technique and the affine transform in conjunction with sum fusion.

3) GSM-AMR: Figure 11 and 12 show the results. The best method is using only the affine transform. The best features are the MFCC, sum fusion and product fusion. Due to interaction, the three best approaches are MFCC with McCree (3.29%), sum fusion with affine (3.26%) and product fusion with affine (3.24%). All three are statistically indistinguishable.

C. Comparison with Testing on Clean Speech

In the case of testing on clean speech, neither signal nor feature enhancement is necessary. Also, there is no statistical difference among the features and fusion methods for both SI and SV systems. The purpose is to compare the performance of the best approaches for each speech coder with the performance on clean speech.

Table III gives the average ISR comparisons for the SI case. There are two approaches that achieve the best average ISR for the G.729 codec. The MFCC feature is selected as the benchmark for clean speech as it achieves the highest average ISR. The best approach for each codec is individually compared to the test case of clean speech only. Therefore, a two sample statistical t-test with a 5% significance level

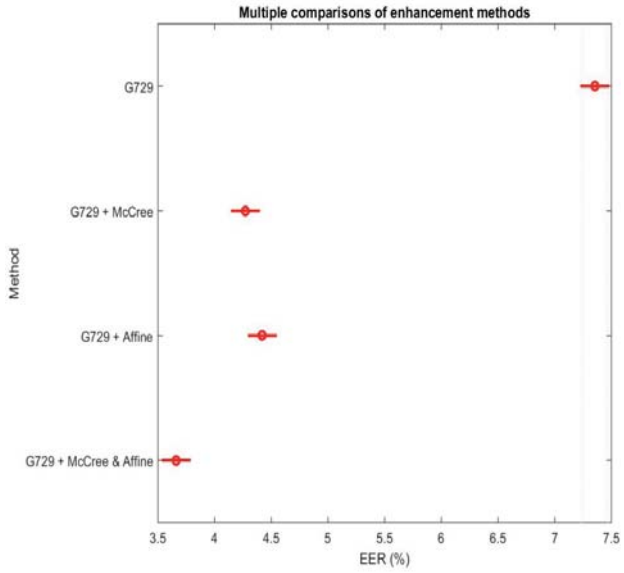


Fig. 9. Speaker verification: Comparison of the methods (G.729).

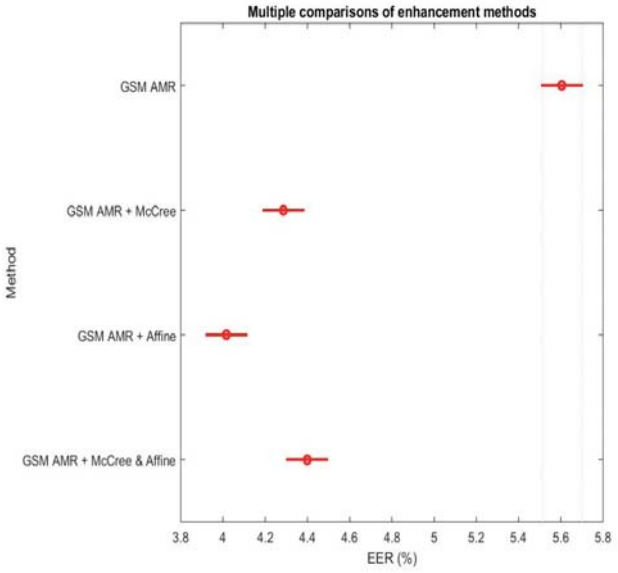


Fig. 11. Speaker verification: Comparison of the methods (GSM-AMR).

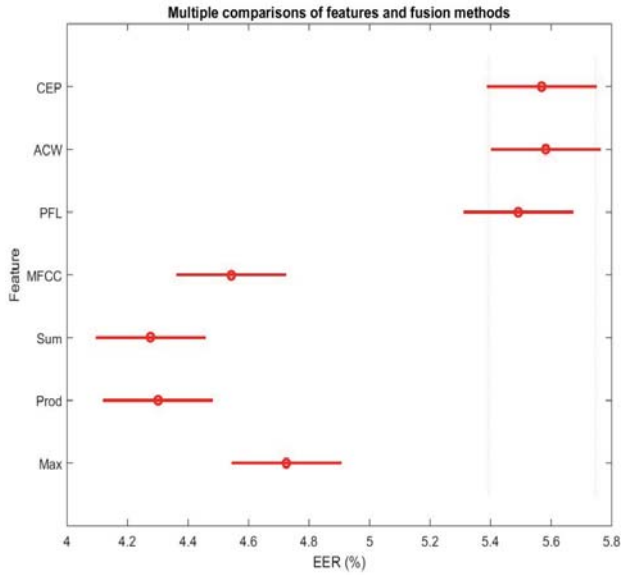


Fig. 10. Speaker verification: Comparison of the features (G.729).

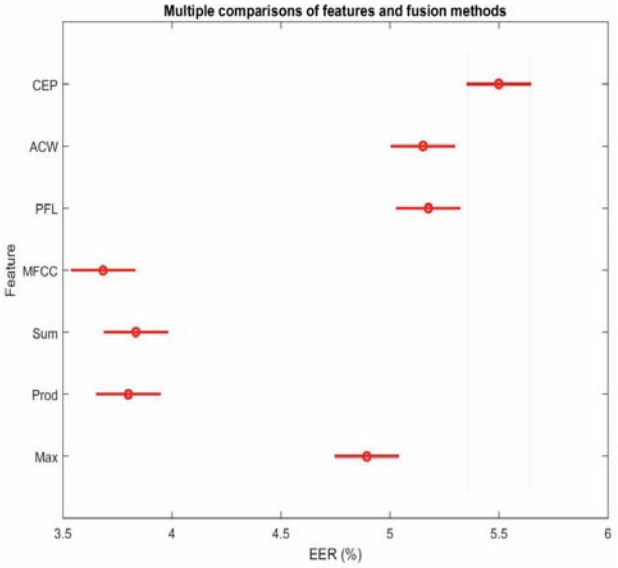


Fig. 12. Speaker verification: Comparison of the features (GSM-AMR).

and unequal variances is performed to determine if the performance on clean speech is significantly better than the technique used for each codec. The test is based on the 10 trials that are performed for each experiment. Table III gives the obtained p-values. Although the methods have mitigated the train/test mismatch and led to a substantial performance improvement, the low p-values indicate that the ISR values are not statistically comparable to that of clean speech.

Table IV gives the average EER comparisons for the SV case. Product fusion is selected as the benchmark for clean speech as it achieves the lowest average EER. Again, the best approach for each codec is individually compared to the test case of clean speech only using a two sample statistical t-test with a 5% significance level and unequal variances. Again, the methods mitigate the train/test mismatch but are not statistically comparable to that of clean speech.

V. Summary and Conclusions

Both the signal (McCree method) and feature (affine transform) enhancement strategies are highly useful in improving the performance of SI and SV systems that are trained on clean speech and tested on the decoded speech. For the G.723.1 and G.729 codec at the relatively lower bit rates, the combination of the McCree technique and the affine transform in conjunction with a fusion strategy is generally the best approach. For the higher bit rate 12.2 kb/s GSM-AMR coder, using only the affine transform with a fusion strategy is the best approach.

Test Speech	Approach	ISR	p-Value
Clean	MFCC	95.4	
G.723.1	McCree and Affine, Borda Count	87.9	1.6e-07
G.729	McCree and Affine, Borda Count	91.1	6.4e-05
G.729	McCree with MFCC	91.1	1.06e-04
GSM-AMR	Affine transform, Decision Level	89.8	1.52e-07

TABLE III
IDENTIFICATION SUCCESS RATE (%) FOR COMPARISON WITH CLEAN SPEECH

Test Speech	Approach	EER	p-Value
Clean	Product Fusion	2.77	
G.723.1	McCree and Affine, Sum Fusion	4.10	2.3e-05
G.729	McCree and Affine, Sum Fusion	3.13	0.02
GSM-AMR	Affine transform, Product Fusion	3.24	9.9e-04

TABLE IV
EQUAL ERROR RATE (%) FOR COMPARISON WITH CLEAN SPEECH

VI. ACKNOWLEDGEMENT

This work was supported by the National Science Foundation through TUES Type 2 Collaborative Research Grants DUE-1122296 and DUE-1122299 awarded to Rowan University and Tennessee State University respectively.

VII. REFERENCES

1) A. K. Vuppala, K. S. Rao and S. Chakrabarti, "Effect of Speech Coding on Speaker Identification", *Annual IEEE India Conference*, Kolkata, India, December 2010.
2) H. Beigi, *Fundamentals of Speaker Recognition*, Springer, 2011.
3) R. Togneri and D. Pullella, "An overview of speaker identification: Accuracy and robustness issues", *IEEE Circuits and Systems Magazine*, pp. 23–61, June 2011.
4) A. Fazel and S. Chakrabartty, "An overview of statistical pattern recognition techniques for speaker verification" *IEEE Circuits and Systems Magazine*, pp. 62–81, June 2011.
5) T. Kinnunen and H. Li, "An overview of text-independent speaker recognition: From features to supervectors", *Speech Communication*, vol. 52, pp. 12–40, 2010.
6) K. Raval, R. P. Ramachandran, S. S. Shetty and B. Y. Smolenski, "Feature and Signal Enhancement for Robust Speaker Identification of G.729 Decoded Speech", *International Conference On Neural Information Processing*, Doha, Qatar, pp. 345–352, November 2012.
7) A. Moreno-Daniel, B.-H. Juang, J. A. Nolazco-Flores, "Robustness of bit-stream based features for speaker verification", *IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, pp. I-749–I-752, 2005.
8) A. McCree, "Reducing speech coding distortion for speaker identification", *IEEE Int. Conf. on Spoken Language Processing*, 2006.
9) J. Vicente-Pena, A. Gallardo-Antoln, C. Pelaez-Moreno and F. Daz-de-Mara, "Band-pass filtering of the time sequences of spectral parameters for robust wireless speech recognition", *Speech Communication*, Vol. 48, pp. 1379–1398, 2006.
10) A. M. Gomez, A. M. Peinado, V. Sanchez and A. J. Rubio, "Recognition of coded speech transmitted over wireless channels", *IEEE Transactions on Wireless Communication*, Vol. 5, No. 9, pp. 2555–2562, September 2006.
11) T. Ogunfunmi and M. J. Narasimha, "Speech Over VoIP Networks: Advanced Signal Processing and System Implementation", *IEEE Circuits and Systems Magazine*, pp. 35–55, 2012.
12) "ITU-T: Recommendation G.723.1 - Dual rate speech coder for multimedia communications transmitting at 5.3 and 6.3 kbit/s", 1996.
13) P. Kabal, "ITU-T G.723.1 Speech Coder: A Matlab Implementation", Telecommunications and Signal Processing Laboratory, McGill University, 2004.
14) "ITU-T: Recommendation G.729 - coding of speech at 8 kbit/s using conjugate-structure algebraic-code-exited linear prediction (CS-ACELP)", 2007.
15) "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Mandatory speech CODEC speech processing functions; AMR speech CODEC; General description", 2012.
16) T. Hasan and J. H. L. Hansen, "A study on universal background model training in speaker verification", *IEEE Transactions on Audio, Speech and Language Processing*, Vol. 19, No. 7, pp. 1890–1899, September 2011.
17) D. A. Reynolds, T. F. Quatieri and R. B. Dunn, "Speaker verification using adapted gaussian mixture models", *Digital Signal Processing*, Vol. 10, pp. 19–41, 2000.
18) C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.
19) I. T. Nabney, *NETLAB: Algorithms for Pattern Recognition*, Springer, 2002.
20) M. S. Zilovic, R. P. Ramachandran and R. J. Mammone, "Speaker identification based on the use of robust cepstral features obtained from pole-zero transfer functions", *IEEE Transactions on Speech and Audio Processing*, Vol. 6, No. 3, pp. 260–267, May 1998.
21) R. Polikar, "Ensemble based systems in decision making", *IEEE Circuits and Systems Magazine*, Vol. 6, No. 3, pp. 21–45, 2006.
22) F. Rastoceanu and M. Lazar, "Score fusion methods for text-independent speaker verification applications", *6th Conference on Speech Technology and Human Computer Dialogue*, 2011.
23) J. L. Devore, *Probability and Statistics for Engineering and the Sciences*, Brooks/Cole Cengage Learning, 2012.

ISBN: 1-60132-431-6, CSREA Press ©

SESSION

REGRESSION AND CLASSIFICATION

Chair(s)

Diego Galar
Robert Stahlbock
Gary M. Weiss

ISBN: 1-60132-431-6, CSREA Press ©

ISBN: 1-60132-431-6, CSREA Press ©

A Cross-Validation Method for Linear Regression Model Selection

Jingwei Xiong Junfeng Shang
Department of Mathematics and Statistics
Bowling Green State University
Bowling Green, OH 43403, USA

Abstract: In linear regression model setting, motivated by Wasserman and Roeder (2009), we develop a cross-validation procedure for selecting an appropriate model which can best fit the data. In the procedure, we make use of adaptive Lasso method to select the most appropriate model. In the selection of the suitable tuning parameter, the Bayesian Information Criterion (BIC, Schwarz, 1978) is utilized. We conduct the hypothesis testings for the significance of nonzero coefficients of fixed effects to further select the model. A simulation study investigates the effectiveness of performance for the proposed procedure. The simulation results demonstrate that BIC and the adaptive Lasso method can both lower Type I error and false positive rate; they can also improve the test power and the rate of selecting an overfitted model.

Key words: Model selection, adaptive Lasso, linear regression, cross-validation, tuning parameter

1 Introduction

Model selection is to select a subset of candidate variables that will contribute to the prediction of the response variable. The traditional subset method and ridge regression have drawbacks. For the subset method, because we add or drop one variable in the model, the procedure is discontinuous. While losing the unbiasedness, the ridge regression may decrease the mean square error (MSE). Further, the predictors whose coefficients are close to zero are still stand in the model, making the model quite complicated and then it is not easily interpreted.

To avoid the problems in traditional methods, penalized model selection in linear regression has caught enough attention. Following the Lasso method, by Tibshirani in 1996, the SCAD (Fan and Li, 2001) and the adaptive Lasso (Zou, 2006) modified the Lasso L-1 penalty term by using adaptive weights, to give consistent estimator of non-zero sets under mild regularity conditions. Candes and Tao (2007) modified the penalty term and developed the Dantzig selector to achieve the same goal. The sparsity pattern lying in these approaches makes the selected model simpler.

A number of other papers also tackle the model selection problem using the penalized method. In this pool, they are Meinshausen and Bühlmann (2006), Wainwright (2006), Zhao and Yu (2006), Fan and Lv (2008), Meinshausen and Yu (2009), Tropp (2004, 2006), Donoho (2006) and Zhang and Huang (2006).

We wish to employ a penalized method to conduct model selection for achieving a better selection result. The paper of Wasserman and Roeder (2009) has investigated a penalized method for high-dimension variable selection, which inspires us to improve its method. We therefore develop a cross-validation procedure to select nonzero effects in linear regression models, whose goal is to obtain a better model selection result using cross-validation, adaptive Lasso and BIC.

In what follows, the model and assumptions are first introduced. The procedure is depicted in Section 3. Section 4 presents a simulation study. The last section concludes and discusses.

ISBN: 1-60132-431-6, CSREA Press ©

2 The model and assumptions

The fundamental settings are first presented and then the assumptions are interpreted.

2.1 The model

Let $(X_1, Y_1), \ldots, (X_n, Y_n)$ be i.i.d observations from the regression model

$$Y_i = X_i^T \beta + \varepsilon_i, \;\; i = 1, \cdots, n, \tag{2.1}$$

where $\varepsilon_i \sim N(0, \sigma^2), X_i = (X_{i1}, \ldots, X_{ip})^T \in R^p$. Here we have n observations and p potential factors affecting the response variable. Let X represent the design matrix for the model and we denote the jth column of the design matrix using $X_\bullet j = (X_{1j}, \ldots, X_{nj})^T$, and the response vector Y is denoted by $Y = (Y_1, \ldots, Y_n)^T$. Let $D = \{j : \beta_j \neq 0\}$ represent the set of covariates with nonzero coefficients. We assume that X and Y are given and known prior to analysis, but D is unknown.

2.2 The assumptions

Throughout the paper, we have the following assumptions:

(A1) $Y_i = X_i^T \beta + \varepsilon_i$ where $\varepsilon_i \sim N(0, \sigma^2)$, for $i = 1, \ldots, n$.

(A2) The covariates are standardized: $E(X_{ij}) = 0$ and $E(X_{ij}^2) = 1$. Also there exists $0 < B < \infty$ such that $P(|X_{jk}| \leq B) = 1$.

(A1) is required since we are dealing the problems under the normal settings.

(A2) means the data are standardized before proceeding to analysis.

We also require that (1) the number of columns of design matrix X is not far beyond the number of rows; (2) the number of non-zero covariates is bounded by a constant regardless of n, and at least we have one non-zero covariates. (3) We require that the largest eigenvalue of p by p matrix $\frac{1}{n} X^T X$, when n approaches to infinity, is almost bounded. The smallest eigenvalue of $C_1 \log n$ by $C_1 \log n$ matrix of $\frac{1}{n} X_k^T X_k$ is almost greater than a positive value for some $C_1 > 0$. Here X_k denotes a design matrix with k predictors in the model. This condition is added due to some proof requirements.

3 Cross-validation procedure

3.1 The procedure

First, we split the data into three parts, D_1, D_2, D_3. The D_1 is to select the most appropriate model through a penalized method, Lasso or adaptive Lasso; the D_2 is to choose the best tuning parameter for D_1 by means of MSE or BIC; that is, the D_1 and D_2 are utilized in one iteration, and the determination of the tuning parameter will affect the selection of final model. The D_3 is to refine the selected model from D_1 and D_2 by removing the insignificant covariates. Theoretically, the D_3 may not be needed, yet it will fortify the selected model.

For the first part D_1, we fit a series of candidate models, and each model depends on a tuning parameter λ. The whole set of candidate models is denoted as $S = \{\widehat{S}_n(\lambda) : \lambda \in \Lambda\}$. Let us call the best candidate model $\widehat{S}_n$, and it can be shown that $P(D \subset \widehat{S}_n) \to 1$ and $|\widehat{S}_n| = o_P(n)$, where $|\widehat{S}_n|$ represents the number of elements in the set S_n. That means the candidate models are overfitted ones which contain the unknown true model and therefore correspond to a sequence of λ values which can determine such overfitted models. This fact will also be discussed afterwards. We address that this fact that D_1 will choose the overfitted models provides the foundation for the entire procedure and guarantees the validity of tests for D_3 and the test power will not be too small. Moreover, it shows the selected power approaches to 1 as the sample size n goes to infinity, thus we can almost reject all the non-zero coefficients when the data size is large enough.

A penalized method will be applied to D_1 for selecting the overfitted models. Of course, a large sample size will result in a large probability of containing the true model. The Lasso penalized target function to be utilized in D_1 is defined as:

$$\sum_{i=1}^{n} (Y_i - X_i^T \beta)^2 + \lambda \sum_{j=1}^{p} |\beta_j|, \tag{3.1}$$

where Y_i, X_i are the corresponding vector and matrix from D_1. To minimize this function, the Lasso estimator of the β will be obtained, and the model will

therefore be selected. In the selection procedure of using D_1, all the non-zero coefficients are included in the selected models.

We adopt the adaptive Lasso penalized method and it follows a quite similar way as the function in expression 3.1, and has been changed to:

$$\sum_{i=1}^{n}(Y_i - X_i^T\beta)^2 + \lambda\sum_{j=1}^{p} w_j|\beta_j|, \tag{3.2}$$

where $w_j = \frac{1}{|\beta_j|}$. Firstly, we compute the β_{ols} from D_1, and take their reciprocals to have the weight for each β_j as starting values. Analogously, minimizing this function will result in the adaptive Lasso estimator of the β. For each λ value, we will obtain a different selected model.

For functions (3.1) or (3.2), we assign multiple different values of λ in a reasonable interval, and each λ outputs a selected model and then the selected model outputs a criterion. The best model is selected corresponding to the smallest criterion, as shown in what follows. With respect to the reasonable interval of λ, it can be investigated in the simulations and then is determined.

For the second part D_2, the target is to select the best λ. The first selection criterion is the mean square error (MSE) and is also called the loss.

Generally, for an estimator, we want to measure how close it was to the true parameter, thus the concept of loss was brought in. The loss of any estimator $\widehat{\beta}$ is defined as:

$$L(\widehat{\beta}) = \frac{1}{n}(\widehat{\beta} - \beta)^T X^T X(\widehat{\beta} - \beta).$$

The loss measures the "distance" between the statistic and the true parameter, and a smaller loss means our estimate statistic is closer to the parameter. Therefore, we want to have a $\widehat{\beta}$, which can minimize the loss. Meanwhile we can see that each tuning parameter λ exports a $\widehat{\beta}$, and $\widehat{\beta}$ determines the loss, so the loss can be treated as a function of λ. We will denote the loss as $L(\lambda)$. Again and more importantly, we have $P(D \subset \widehat{S}_n(\widehat{\lambda})) \to 1$ where $\widehat{\lambda} = argmin_{\lambda\in\Lambda_n} L(\lambda)$, indicating that by the loss function, the overfit or overspecified models are selected.

However, due to the unknown β, we plug in the $\widehat{\beta}(\lambda)$ to a substitute estimable formula for loss, we then have

$$\widehat{L}(\lambda) = \frac{1}{n}\sum_{X_i\in D_2}(Y_i - X_i^T\widehat{\beta}(\lambda))^2.$$

Corresponding to the "best" selected λ, the estimated loss $\widehat{L}(\lambda)$ is minimized. Note that the estimated loss $\widehat{L}(\lambda)$ behaves similarly as the true loss. This optimal property is shown by Theorem 3.2 in Wasserman and Roeder (2009). Suppose that the $max_{\lambda\in\lambda_n}|\widehat{S_n}(\lambda)| \leq k_n$. Then there exists a sequence of random variables $\delta_n = O_P(1)$ that do not depend on λ or X, such that with probability tending to 1,

$$\sup_{\lambda\in\Lambda_n}|L(\lambda) - \widehat{L}(\lambda) - \delta_n| = O_P(\frac{k_n}{n^{1-c_2}}) + O_P(\frac{k_n}{\sqrt{n}})$$

Note that if k_n is a good sequence which makes $\frac{k_n}{n^{1-c_2}}, \frac{k_n}{\sqrt{n}}$ converge to 0, the difference between $\widehat{L}(\lambda)$ and $L(\lambda)$ can be approximated by a random variable δ, which is bounded in probability. Based on this theorem, the estimated loss $\widehat{L}(\lambda)$ performs as the true loss. For the true loss, the ovefitted models are selected, which is identical to the selecting procedure of D_1 in that the penalized method will also choose the overfitted models which include the true parameters or true model.

Now we remark on the procedure using D_1 and D_2. In D_1, the penalized Lasso method will choose the overfitted models; in D_2, the loss function MSE will also choose the best λ which agrees with an overfitted model. So it is convincing to develop the cross-validation for D_1 and D_2, and we select the λ which minimizes the $\widehat{L}(\lambda)$ for the data D_2. After the cross-validation procedure using D_1 and D_2, the "best" model which is overfitted will be selected.

The criterion we take for selecting the best λ is BIC, and it is written as

$$\text{BIC} = n\log\hat{\sigma}^2 + k\log n,$$

where $\hat{\sigma}^2$ is computed by $\frac{1}{n}(Y - X\widehat{\beta}_{ols})^T(Y - X\widehat{\beta}_{ols})$, and k is the number of the estimated parameters in the corresponding model. Note that the constant in BIC is ignored here. Based on the model selected

in D_1 for each λ, we can estimate the least squares estimate of β, $\widehat{\beta}_{ols}$ in D_2, and n is the sample size in D_2, BIC can therefore be calculated in D_2. We select the λ which minimizes the $\widehat{L}(\lambda)$ or BIC for the data D_2. After the best λ is selected by minimizing the criterion MSE or BIC, by functions in (3.1) or (3.2), the best model will be selected.

Note that as a model selection criterion, MSE is efficient yet not consistent; whereas BIC is consistent, which means that BIC will select the true model with probability 1 as the sample size increases to infinity. We therefore expect that BIC can perform better in selecting the tuning parameter λ.

For high-dimensional model selection, it requires the property that $|\widehat{S_n}| = o_p(n)$, which means when n approaches to ∞, the number of selected non-zero regression coefficients is much smaller than n in probability. This property ensures that the subsequent steps of the procedure can function well even in high-dimensional model selection.

We remark that in the previous procedure, we adopt the adaptive Lasso and BIC for the choice of proper model and tuning parameter. Although the method in Wasserman and Roeder (2009) is effective in high-dimensional model selection, the adaptive Lasso can perform well only if the matrix $X^T X$ is of full rank, so we can assume $p < n$ to facilitate the implementation of adaptive Lasso.

For the third part D_3, we will finalize the selected model. In fact, using D_3 is one step which is not required for model selection. However, this step can always improve the rate of selecting the correct model. We thus consider it as an effective supplementary step to the cross-validation procedure.

For the best model from the previous steps, we can find its corresponding least squares estimate $\widehat{\beta}$ by using the data D_3. Then for each coefficient in the $\widehat{\beta}$, we use the t-test to decide which coefficients will be included, and the final non-zero set is given by:

$$\widehat{D}_n = \{j \in \widehat{S}_n : |T_j| > c_n\},$$

where T_j is the t-statistic, $c_n = t_{\alpha/2m, n-m}$ or $c_n = z_{\alpha/2m}$ and $m = |\widehat{S}_n|$.

To be more specific, the estimated covariance matrix can be computed by $\widehat{\sigma}^2 (X_M^T X_M)^{-1}$, where $\widehat{\sigma}^2 = \frac{SSE}{n^* - p^*}$. Here, X_M represents the design matrix for the best model, n^*, p^* represent its numbers of rows and columns respectively in D_3. The α is divided by $2m$ due to the Bonferroni correction, and m is the number of t tests.

Theorem 4.1 in Wasserman and Roeder (2009) states $P(D \subset \widehat{S}_n(\widehat{\lambda})) \to 1$ where $\widehat{\lambda} = argmin_{\lambda \in \Lambda_n} \widehat{L}(\lambda)$. This theorem makes sure that the subset selected by the cross-validation procedure can cover all the non-zero coefficients with probability 1. Consequently, the best model selected from this procedure is overfitting, and the hypothesis testings in D_3 for t-tests are not biased. When the criterion BIC is utilized, as the sample size n goes to infinity, we can have $P(D = \widehat{S}_n(\widehat{\lambda})) \to 1$ where $\widehat{\lambda} = argmin_{\lambda \in \Lambda_n} \mathrm{BIC}(\lambda)$. As mentioned earlier, because of the consistency of BIC in selecting models, it may be expected that using BIC instead of $\widehat{L}(\lambda)$, the model selection will result in a better implementation.

Theorem 4.2 asserts the consistency of the procedure. In particular, let $\alpha_n \to 0$, and$\sqrt{n}\alpha_n \to \infty$, then we have $P(\widehat{D}_n = D) \to 1$, which indicates that as the sample size n goes to infinity, this cross-validation procedure will choose the true model with probability 1, and we can achieve a consistent estimate of the non-zero subset by utilizing this cross-validation procedure.

3.2 Type I error and test power

The goal of the paper is to derive a procedure $\widehat{D}_n$, such that:

$$\limsup_{n \to \infty} P(\widehat{D}_n \subset D) \geq 1 - \alpha.$$

The formula above shows that the probability of selecting "false positive" variable is controlled by α. In other words, the probability of rejecting zero covariates is less than α, thus the Type I error is controlled. When n is large enough, the inequality is satisfied for whatever positive α. In a sense, we can let the α approach to 0, ensuring that the selected subset does not involve any zero covariates. Note that as n goes to infinity, the asymptotic results will be used in the derivation.

On the other hand, we wish the procedure can take nontrivial power, thus we still have chance to reject

the nonzero covariates. We will compare the rejecting power of this procedure utilizing adaptive Lasso and BIC with that in Wasserman and Roeder (2009) utilizing Lasso and MSE in a simulation study in the next section.

For any test, we have two major concerns, the error and the power. The type I error rate regarding the estimated non-zero subset is defined as:

$$q(\widehat{D}_n) = P(\widehat{D}_n \cap D^c \neq \emptyset).$$

The power is defined as:

$$\pi(\widehat{D}_n) = P(D \subset \widehat{D}_n).$$

We can see the Type I error actually means the probability of rejecting any zero-coefficients. The power actually means the probability of rejecting all the non-zero coefficients. Our target is to keep a low error rate and meanwhile to maximize the power. Regarding the power presented in the tables of next section, we use the average power which is defined as:

$$\pi_{av} = \frac{1}{s}\sum_{j\in D} P(j \in \widehat{D}_n),$$

where s is the number of non-zero coefficients.

4 A simulation study

4.1 The simulation settings

First, the data are generated from the true models in equation (2.1). For the design matrix X, each element x_{ij} is generated from $N(0, 2)$, the normal distribution with mean 0 and variance 2. For the error term ϵ, each element ϵ_j is simulated from $N(0, 1)$. For the regression coefficients vector β, we have two true models.

Model (a): $\beta = \text{rep}(0, 20)$, which contains 20 predictors with zero coefficients.

Model (b): $\beta = \text{c}\ (9 : 1, \text{rep}(0, 11))$, which contains 9 non-zero coefficients and 11 zero coefficients.

For the true models, we have dimension $p = 20$. Utilizing the generated X, β and ϵ, we can compute the response vector Y under the true model.

For each of the two true models, we generate 1000 replicates, and for each replicate, we implement the cross-validation procedure. As the result, we will have the coefficient selection for 1000 replicates and $p = 20$ parameters either zero or nonzero.

Prior to analysis, the covariates are scaled with mean 0 and variance 1. The tests are performed using one third of the data for each of the 3 stages of the procedure. The level α is set as 0.05, and we want to compare the 3 approaches mentioned in stage one and detect which level $\alpha = 0.05$ test gives the greatest power.

As introduced for the procedure, we split the data (one replicate) into 3 parts, D_1, D_2 and D_3.

For D_1, we choose a sequence values of λ from 0.01 to 0.4 (reasonable interval for λ). In fact, there are 70 values with equal width. For each λ, since the Y_i, X_i are known, this is a 20 dimensional function of β, we use the "optim" function in R to minimize the target function (3.1) or (3.2). The problem is that the numerical solution can never be 0 for β_j, thus we set the boundary value as 0.03. If β_j is less than 0.03, then it could be treated as 0. In this way, for each λ, a candidate model is selected from D_1.

For D_2, we use the candidate model from D_1 to compute the criterion (MSE or BIC). Computing MSE or BIC, we record the candidate model corresponding to the minimum MSE or BIC value, which is considered as the best model.

For the model selected from D_2, we compute the β_{ols} regarding D_3, the covariance matrix of $\hat{\beta}$ is $\hat{\sigma}^2(X^TX)^{-1}$. Taking the diagonal elements of the covariance matrix, we have the estimated variance regarding each $\hat{\beta}_j$. Then we compute the t-test statistic of each coefficient by $\frac{\hat{\beta}_j}{sd(\hat{\beta})}$, if it is greater than $T_{\alpha/2m,n-m}$, we reject and consider that coefficient as non-zero in the final model. Note that m is the number of coefficients in the best model from the previous steps. Since $n-m$ is very large, we may approximate the t-value by z-value.

4.2 The simulation results

We repeat the procedure for 1000 times and organize the results for a replicate in the format of a vector from the simulations, and all the results are stored in a 1000×20 matrix. For each replicate, we use an indicated vector (length is 20) to record the

Table 1: Selection results for model (a)

Model (a)	(c)	(d)	(e)
Lasso. MSE	0.0023	0.0360	1.0
Lasso. BIC	0.0017	0.0320	1.0
A.Lasso. MSE	0.0021	0.0340	1.0
A.Lasso. BIC	0.0016	0.0320	1.0

Note: (c) = FPR, (d)=Type I error, (e)=Overspecified.

Table 2: Selection results for model (b)

(b)	(c)	(d)	(e)	(f)
(1)	0.0027	0.0300	0.9550	0.9950
(2)	0.0027	0.0275	0.9825	0.9980
(3)	0.0016	0.0175	0.9650	0.9960
(4)	0.0020	0.0225	0.9750	0.9972

Note: (b) = Model (b), (c) = FPR, (d)=Type I error, (e)=Overspecified, (f)=AV-Power, (1)= Lasso with MSE, (2)= Lasso with BIC, (3)= adaptive Lasso with MSE, (4)= adaptive Lasso with BIC.

selection result. For instance, if an indicated vector is (0,0,0,1,0...,0), it suggests only the 4th coefficient are kept in the final model.

Since true model (a) is a null model, the expected vector of estimated parameters is $(0, 0, \cdots, 0)$. If the ith predictor is not selected in the model, the ith element is denoted as 0; otherwise, keep it as 1. We compute the column means, which is the rejection rate of each regression coefficient, which is also the false positive rate. Since there are no non-zero coefficients, the power is always 0. The type I error occurs when a result vector contains not all 0.

For true model (b), the expected vector of estimated parameters is $(1, ...1, 0, 0, \cdots, 0)$, where the first 9 elements are 1 and the rest are 0. Similarly, we can compute the false positive rate of each regression coefficient. The average power is to average over the rejection rates in the first 9 columns, which are the first 9 column means. Type I error appears when the last 11 elements of a result vector are not all 0.

For model (a), Type I error occurs when at least one of the coefficients are not zero, the false positive rate is computed as the rate of 1 occurring in the result matrix. The rate of overspecified models is always 1 and the power does not exist for the model (a). Similarly, we can output everything for the model (b). The results are listed in Tables 1 and 2.

Note that "FPR" represents the false positive rate; "overspecified" column inputs the rate that the selected model contains the true model; "AV-Power" represents the average power for all the t-tests in D_3.

From the numbers in Table 1, we can observe that for the model with all zero coefficients, employing BIC for selecting the appropriate tuning parameter can lower the false positive rate and the probability of Type I error. The adoption of the adaptive Lasso for selecting the appropriate model can decrease the above two rates as well.

Table 2 shows that BIC and the adaptive Lasso can not only lower the false positive rate and the probability of Type I error, but also increase the rate of selecting a model containing the true model and test powers.

The simulation study shows that BIC and the adaptive Lasso are an optimal choice for the cross-validation method in model selection.

5 Concluding remarks

Based on the method in Wasserman and Roeder (2009), we adopt the adaptive Lasso and BIC for selecting a model and for selecting the tuning parameter respectively to develop a cross-validation model selection procedure in linear regression model setting.

The simulation results demonstrate that BIC and the adaptive Lasso method can both lower Type I error and false positive rate, and meanwhile they can also increase both the test power and the rate of selecting a model containing the true model.

This paper is the launch of further research. Starting from the procedure in this paper, we will extend similar methodology to generalized linear, lin-

ear mixed, and generalized linear mixed models with solidified proofs. Of course, the future research is confronted with quite a few challenges. For instance, in the step of selecting a model, handling the random effects using the penalized method is appealing. Regarding random effects selection, we may consider using the EM algorithm. Treating the random effects as unobserved data, we may compute the conditional expectation of likelihood given the random effects, which is considered as E-step. Other than that, different papers propose different penalty terms, which makes the target expectation different, then M-step is to maximize the target function.

The other idea is to estimate the mixed effects and random effects separately. Moreover, we can replace the unknown covariance matrix by some simple matrix, such as $\log n \times I$, which significantly simplify the penalized likelihood.

In addition, we will consider the model selection with missing values. Missing values occur commonly, effectively coping with missing values therefore plays an important role in model selection literature. We may explore imputation and bootstrapping for missing values.

It is expected that the bootstrap method will improve the test power. However, the necessary bootstrapping theories must be steadily justified.

We can also investigate leave-one-out cross-validation to make this procedure more effective.

References

[1] Candes, E. and Tao, T. (2007). The Dantzig selector: Statistical estimation when p is much larger than n. *The Annals of Statistics*, pp. 2313-2351.

[2] Donoho, D. (2006). For most large underdetermined systems of linear equations, the minimal l1-norm near-solution approximates the sparsest near-solution. *Comm. Pure Appl. Math.* **59** 797-829.

[3] Fan, J. and Li, R. (2001). Variable selection via nonconcave penalized likelihood and oracle properties. *Journal of the American Statistical Association* **96**, 1348-1360.

[4] Fan, J. and Lv, J. (2008). Sure independence screening for ultra-high-dimensional feature space. *J. Roy. Statist. Soc. Ser. B* **70** 849-911.

[5] Meinshausen, N. and Bühlmann, P. (2006). High dimensional graphs and variable selection with the lasso. *Ann. Statist.* **34** 1436-1462.

[6] Meinshausen, N. and Yu, B. (2009). Lasso-type recovery of sparse representations of high dimensional data. *Ann. Statist.* bf 37 246-270.

[7] Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics* **6**, 461-464.

[8] Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, Series B **58**, 268-288.

[9] Tropp, J. A. (2004). Greed is good: Algorithmic results for sparse approximation. *IEEE Trans. Inform. Theory* **50** 2231-2242.

[10] Tropp, J. A. (2006). Just relax: Convex programming methods for identifying sparse signals in noise. *IEEE Trans. Inform. Theory* **52** 1030-1051.

[11] Zhang, C. H. and Huang, J. (2006). Model selection consistency of the lasso in high-dimensional linear regression. *Ann. Statist.* **36** 1567-1594.

[12] Zhao, P. and Yu, B. (2006). On model selection consistency of lasso. *J. Mach. Learn. Res.* **7** 2541-2563.

[13] Zou, H. (2006). The adaptive lasso and its oracle properties. *Journal of the American Statistical Association* **101**, 1418-1429.

[14] Wainwright, M. (2006). Sharp thresholds for high-dimensional and noisy recovery of sparsity. Available at arxiv.org/math.ST/0605740.

[15] Wasserman, L and Roeder, K. (2009). High-Dimensional variable selection. *The Annals of Statistics* **37, No 5A** pp. 2178-2201.

Convolutional Neural Net and Bearing Fault Analysis

Dean Lee, Vincent Siu, Rick Cruz, and Charles Yetman
SPAWAR Systems Center Pacific
San Diego, CA, USA
Email: {dean.lee1,vincent.siu,rick.cruz,charles.yetman}@navy.mil

***Abstract*—There has been immense success on the application of Convolutional Neural Nets (CNN) to image and acoustic data analysis. In this paper, rather than preprocessing vibration signals to denoise or extract features, we investigate the usage of CNNs on raw signals; in particular, we test the accuracy of CNNs as classifiers on bearing fault data, by varying the configurations of the CNN from one-layer up to a deep three-layer model. We inspect the convolution filters learned by the CNN, and show that the filters detect unique features of every classification category. We also study the effectiveness of the various CNN models when the input signals are corrupted with noise.**

***Index Terms*—Classification algorithms; multi-layer neural network; signal analysis; fault diagnosis.**

I. Introduction

Anomaly detection from vibration signals has traditionally used sophisticated methods derived from signals processing, usually relating to power spectrum analysis or wavelet-based methods. While these tried-and-true techniques have become an important tool set for the prognostics and health management community, they require extensive expertise, expensive laboratory test rigs, and experiments to properly design robust fault detection algorithms. Moreover, these techniques are sensitive to changes in physical characteristics of the vibrating component, e.g. a change in the manufacture of the component may dramatically alter its vibration signatures. The laboratory tests must be re-run with the new components and the detection algorithms are then re-calibrated to account for the changes. This is a major disadvantage for physics-based models.

Advances in machine learning have enabled many data-driven techniques to overcome the drawbacks of traditional signals analysis techniques. In particular, neural-net-based methods have caught the attention of many communities in recent years due to their automated process of learning latent concepts and their ability to transform the learned models into powerful and highly accurate classification algorithms. The Convolutional Neural Network (CNN) is one such neural network architecture that has shown immense possibilities in image processing and audio processing [1]–[3].

Most of the recent applications of neural networks in vibration analyses have focused on denoising autoencoders, which is a type of neural network that is built with layers of overcomplete and undercomplete hidden neurons to extract the latent features from noisy data [4]–[6]. In this paper, we explore the use of CNN to classify different types of signals by using open rolling bearing vibration data sets [7,8]. In particular, we show that the convolution filters learned by the CNN enable the model to achieve high classification accuracy. We also experiment with a deep CNN architecture study the robustness of deep models in the presence of noisy signals.

II. Related Work

The most basic technique to detect bearing vibration is to establish a baseline using the Root Mean Square (RMS) or Crest Factor of the vibration levels of the bearing housing; a fault is detected when the vibration exceeds some threshold. Clearly this technique is overly simplistic and does not distinguish among different classes of faults.

More often, Fourier-analytic based methods are the tools of choice when it comes to bearing fault diagnosis. For example, time waveform analysis and frequency spectral analysis rely on extensive use of the Fourier transform to shuttle the signals between the time and frequency domains to detect faults; in particular, harmonic analysis is a cornerstone of these type of methods [9].

High frequency detection is yet another method that's used to detect faults in rotating machinery. The idea is that cracking or abrasive wear of the rotating element generates high stress waves, which can then be used as signatures to detect faults [10].

On the other hand, enveloping [11] is a sophisticated technique that was developed to uncover low frequency fault signals, which relies on determining the unique pass frequencies at which various faults can happen. Most enveloping methods seek to determine the optimal window to discern particular faults.

Wavelet-based methods are also popular. While they address many of the short-comings of Fourier-analytic methods, wavelet methods rely on finding the proper bases to be effective [12]. Still, there were some successes using wavelet-based methods combined with machine learning techniques [13].

All of the above methods require experts who are well-versed in signals analyses as well as interpretation of physical signals emanating from the machinery. Moreover, these methods are highly dependent on the physical characteristics of the machinery; for example, if the manufacturing processes of the rotating parts were to change, the fault detection algorithm would need to be re-tuned to account for the new vibration

ISBN: 1-60132-431-6, CSREA Press ©

signatures. Recent advances in data-driven classification techniques have removed the need for such retooling. Instead, the algorithms can learn from the data and adapt to changing conditions.

One such technique that has shown great promise is through the use of deep learning, in particular the usage of autoencoders [4]–[6]. Deep autoencoders are able to pick up latent signals by alternating undercomplete and overcomplete hidden layers. However, autoencoders are relatively simple constructs that do not account for the complexity of vibration analysis.

The Convolutional Neural Network, however, can learn optimal convolutional filters that minimizes an error criterion, and therefore is the ideal candidate for automated bearing fault detection. Similarly in automatic speech recognition (ASR), a CNN architecture using raw accoustic signals has already been shown competitive to standard spectral approaches [3]. In this paper, we use the CNN and also apply a deep-learning approach to bearing faults detection on single- and dual-channel raw vibration signal data sets.

III. Convolutional Neural Network

The Convolutional Neural Network is an architecture made up of three distinct layers: an input layer, a convolutional layer, and a pooling layer. The input layer is made up of neurons that hold data values; optionally, preprocessing of the data values may be done prior to the input.

The parameter κ specifies the number of *feature maps* in the convolutional layer. In the case of one dimensional input, such as vibration signals, κ is exactly the number of filters of which the CNN will learn. Let f and g be two functions, then the discrete convolution of f and g is given by

$$(f * g)[n] \equiv \sum_{m=-\infty}^{\infty} f[n-m]g[m].$$

In the CNN, f is the input layer, and g is one of the κ filters which the CNN will optimize to an objective function during the learning process.

The pooling layer downsamples the output from the convolutional layer. Most common strategies are min-, max-, and average-pooling [14].

A *deep* CNN is an architecture where multiple units of CNN are stacked on top of each other, such that the output from the pooling layer of the CNN below becomes the input for the current CNN. Deep learning has been shown to successfully tackle many problems [2,15,16].

However, deep architectures are notoriously difficult to train [17]. Not only does the network take a lot longer to train than other learning techniques, but often times the neural networks have many more hyper parameters for which to optimize and are very resource intensive. Thus, it often takes many trial and error before a suitable model is found. Part of the contributions of this paper is to determine if a deep CNN architecture is suitable for vibration analysis.

Note that it is common practice, especially with vibration signals analysis, to precondition the data to aid the learning process. Techniques such as Fourier analysis, Wavelet analysis, or dimensionality reduction are often applied to extract features or denoise the data before it is fed into the learning algorithm. In this paper, we also show that the CNN with minimal data conditioning can achieve high classification accuracy and that no sophisticated preprocessing is required.

IV. Experiments and Analysis

Methodology

The proposed architecture shown in Figure 1 consists of two main components: filtering layer(s) followed by a classification layer. The input is standardized to zero mean and unit variance, which will improve training performance [18]. The hidden neurons at each layer are ReLU units, which were empirically shown to be robust [19].

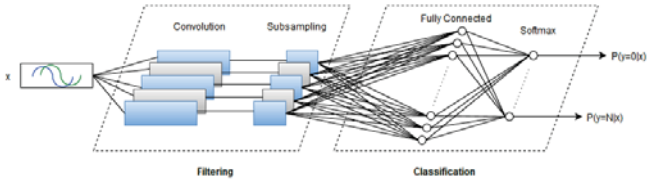


Fig. 1. Implementation of CNN classifier

The filter layers consists of adjacent convolutional and subsampling layers stacked 1 to N times. The convolutional layer convolves the input with a learnable kernel or filter and puts it through a nonlinear ReLU activation function to form an output map. Multiple filters results in the output map having a depth equal to the number of filters. These output maps will show the activation spots when the filters see specific features in the input. The subsampling layer reduces the spatial size of the output map, while the depth dimension remains the same. This is to reduce the amount of parameters in the network and minimize overfitting. Due to the reduced parameters, the computation time will also be reduced.

The output is then fed into two additional layers to perform classification. The first is a fully connected layer composed of sigmoid units that has dropouts enabled to minimize overfitting. The top layer has a softmax function, which will output a conditional probability for each class. A gradient descent optimizer is used to minimize the cross-entropy error when training the system.

We take two publicly available bearing vibration data sets [7,8], which are single- and dual-channel data sets, respectively, and study the effects of tuning various CNN parameters. There are literature that study various methods on extracting bearing fault information from these public data sets [6,11] and self generated data sets [20], but to the best of our knowledge there does not exist a CNN-based method yet for bearing fault analysis that operates on raw vibration signals. Additionally, our experiments test the robustness of CNN when the vibration signals come from multiple channels. We then extend the base CNN configurations to a deep architecture and study the efficacy of deep-CNN on these data sets.

Also, it was found in [21,22] that the injection of corrupted values into the training set can add robustness to the network.

In our experiments, we extend this idea further by randomly corrupting the training data and the test data with Gaussian noise. There are three objectives: 1) when training data is corrupted with noise, the accuracy of the final model should be higher than the model trained without noise corruption [21]; 2) when the test data is corrupted with noise, the robustness of the model trained with uncorrupted data is tested; and 3) when both the training and test data are corrupted with noise, the resiliency of the model against noisy data is tested.

A. Data Sets

The MFPT data set [7] is made up of three sets of bearing vibration data: 1) a baseline set, sampled at 97656 Hz for 6 seconds in each file; 2) an outer race faults set, sampled at 48828 Hz for 3 seconds in each file; and 3) an inner race faults set, sampled at 48828 Hz for 3 seconds in each file. The data points come from a single-channel radial accelerometer. The baseline set was downsampled to 48828 Hz to match the other fault sets. There are additional data files included in the MFPT data set, but they are not used in the experiments.

The Case Western data set [8] is made up of ball bearing data collected from motor bearings that are either normal or faulty. The testbed consisted of a 2 hp motor, torque transducer/encoder, and dynamometer. Accelerometers sampling at 12 kHz were attached to the drive end (DE) and fan end (FE) of the motor housing to measure the vibration. Vibration data was recorded with various engine loads (0 to 3 hp) running at 1700 RPM, motor shaft bearings with faults ranging in depth (none, 0.007, 0.014, 0.021 inches), and fault orientation (inner race, rolling element, and outer race). The outer race fault is centered relative to the load zone positioned at 6:00. The baseline test had 240k data points, while each fault test had 120k data points from each DE and FE sensor.

B. MFPT Data

The data set is divided into three categories: baseline condition, outer race fault conditions, and inner race fault conditions. The data is normalized to 0 mean and unit variance. The goal of the classifier is to correctly categorize the input data to the three conditions.

A single layer CNN was trained with variable number of filters, from 1 to 50 filters. See Figure 2(a) for partial results. We find that the number of filters required to learn the MFPT signals is small; at 5 filters the model achieved roughly 98.2% accuracy. A single layer CNN with up to 50 filters was tested (not shown) and we confirmed that the accuracy does not improve much beyond the 5-filter model. The 5-filter CNN model was then extended to 2 and 3 layers, but we found that the deep CNN model did not improve upon the accuracy of the single layer model.

The learned filters are taken from the single layer CNN and applied to the raw input signals. See Figures 3 and 4 for a comparison between the frequency spectrum of the raw input data and the results of applying convolution using the learned filters from the CNN to the raw input data. The learned filters seemed to pick up distinctive features from each data set which help the classifier categorize the data with high accuracy.

		ν					
		0.05	0.1	0.15	0.2	0.25	0.3
η	0.05	0.9801	0.9777	0.9695	0.9694	0.9584	0.9558
	0.1	0.9725	0.9723	0.9731	0.9664	0.9643	0.9575
	0.15	0.9504	0.9619	0.9658	0.9656	0.9596	0.9514
	0.2	0.9024	0.9407	0.9532	0.9557	0.9508	0.9489
	0.25	0.8379	0.9058	0.9296	0.9397	0.944	0.9418
	0.3	0.7635	0.8486	0.893	0.9025	0.925	0.929

TABLE I
MFPT DATA CLASSIFICATION ACCURACY, 1 LAYER CNN, CORRUPTED TRAINING AND TEST SET

		ν					
		0.05	0.1	0.15	0.2	0.25	0.3
η	0.05	0.9818	0.9826	0.9796	0.9795	0.9656	0.96
	0.1	0.9715	0.9769	0.9688	0.9666	0.9731	0.9579
	0.15	0.9532	0.9695	0.9719	0.9616	0.9707	0.9625
	0.2	0.9117	0.9473	0.9602	0.9634	0.9611	0.9581
	0.25	0.8525	0.901	0.9371	0.9498	0.9483	0.9444
	0.3	0.7859	0.8321	0.8919	0.928	0.9304	0.9321

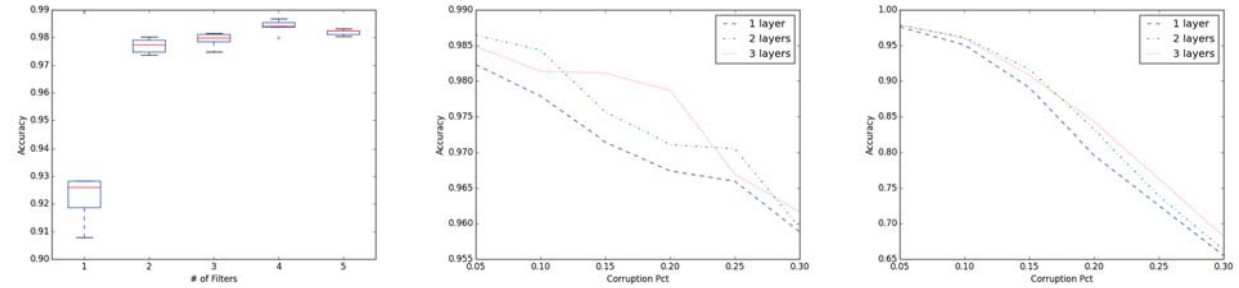
TABLE II
MFPT DATA CLASSIFICATION ACCURACY, 2 LAYER CNN, CORRUPTED TRAINING AND TEST SET

Noise Analysis: We examine the impact of deep architectures when noise is present in the vibration signals. For this analysis, the input signal is randomly corrupted with Gaussian noise. The corruption can happen at 1) the training set, 2) the test set, and 3) both the training and test sets. The corruption parameter ν is the proportion of the training set that is randomly corrupted, and ν ranges from 0.05 to 0.3. The corruption parameter η is the proportion of the test set that is randomly corrupted, and η ranges from 0.05 to 0.3.

Figure 2(b) shows the results of deep CNN configurations against various values of ν. Somewhat surprisingly, the performance of 3-layer CNN is relatively unstable, although its performance at $\nu = 0.2$ is still around 98%, handily beating the other deep-CNN models. Overall, it can be seen that the addition of random perturbation to the training set increases the robustness of the models at all configurations and slightly improves upon the 5-filter single layer CNN classification accuracy.

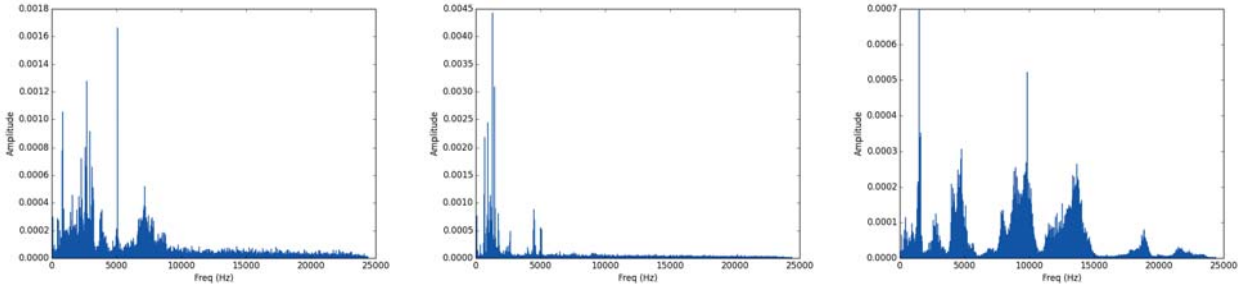
Figure 2(c) shows the results of varied number of layers of CNN that is trained with uncorrupted data, but is tested with corrupted data. It shows that when the input is corrupted up to $\eta = 0.1$, the deep CNN still has relatively high accuracy. However, beyond $\eta = 0.1$ the model begins to lose fidelity quickly. This analysis shows that while a deeper architecture proves to be more resistant against corrupted signals, it is only effective up to a certain point. Furthermore, the deep architectures show the same downward trend as the 1-layer CNN.

Finally, the effects of varying η and ν are tested against the deep CNN architecture. See Tables I, II, and III for results. As expected, deeper architectures with higher values of ν (up to a certain point) seems to handle signal noise better.



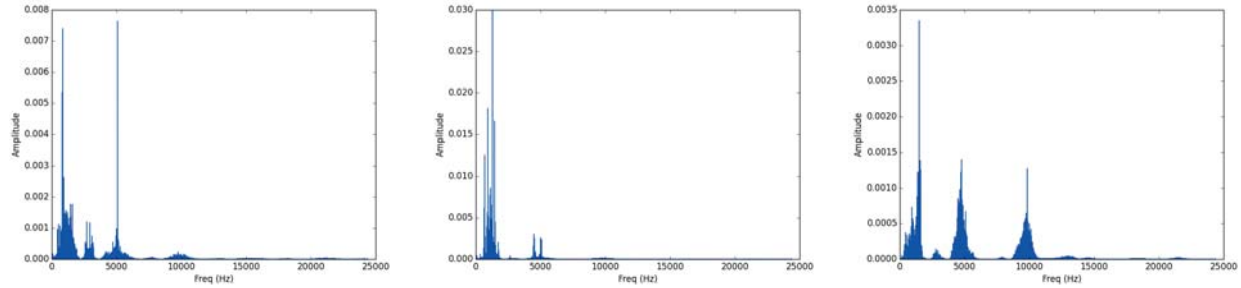
(a) 1-Layer CNN with variable number of filters (b) Deep CNN with Corrupted Training Set (c) Deep CNN with Corrupted Test Set

Fig. 2. MFPT Data, CNN Performance



(a) Raw Baseline (b) Raw Outer Fault (c) Raw Inner Fault

Fig. 3. MFPT, Raw Input Data



(a) Smoothed Baseline (b) Smoothed Outer Fault (c) Smoothed Inner Fault

Fig. 4. MFPT, Smoothed Data

		ν					
		0.05	0.1	0.15	0.2	0.25	0.3
η	0.05	0.9832	0.9781	0.9773	0.9716	0.9712	0.9522
	0.1	0.9716	0.9769	0.9783	0.9757	0.9676	0.9617
	0.15	0.9496	0.9697	0.9713	0.974	0.9692	0.9603
	0.2	0.9074	0.9487	0.9659	0.9678	0.9615	0.9584
	0.25	0.8782	0.9119	0.938	0.9516	0.9535	0.9545
	0.3	0.7635	0.8575	0.8955	0.9273	0.9318	0.9409

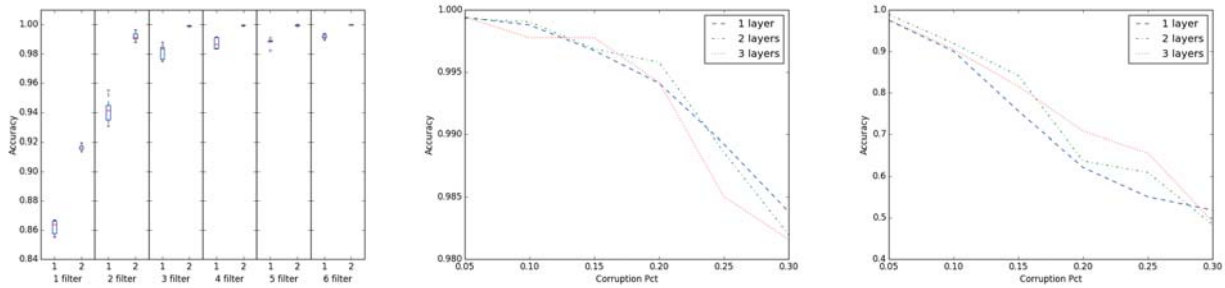
TABLE III
MFPT DATA CLASSIFICATION ACCURACY, 3 LAYER CNN, CORRUPTED TRAINING AND TEST SET

C. Case Western Data

The data set is divided into four categories: baseline condition, outer race fault conditions, rolling element (ball) fault conditions, and inner race fault conditions. The data is standardized to 0 mean and unit variance. The goal of the classifier is to correctly match the input data to the four conditions.

A single layer CNN was trained with a variable number of filters (1 to 10 filters) and data channels (single- or dual-). See Figure 5(a) for a partial graph of the results. We found that the amount of filters to achieve 99% accuracy was small, 6 filters for single-channel and 3 filters for dual-channel. The single-channel exhibited a higher amount of variance and required 10 filters to match the accuracy of the dual-channel. When the single layer 3-filter dual-channel model was extended to 2 or 3 layers, there were negligible improvements.

The frequency spectrum of input data from each category convolved with the learned filters in a single layer CNN



(a) 1-Layer CNN with variable number of filters (b) Deep CNN with Corrupted Training Set (c) Deep CNN with Corrupted Test Set

Fig. 5. Case Western Data, CNN Performance

can be seen in Figures 6 and 7. Only one filtered result with the highest frequency spectrum amplitude is displayed for the baseline and each fault category. It can be seen that a large amount of high frequency energy is produced due to ball bearing defects, resulting in the typical bearing fault frequencies (BPFO, BPFI, BSF) being modulated by a high frequency carrier signal [23]. The figures show that the CNN has learned filters to extract a frequency spectrum profile of the various bearing conditions regardless of high frequency behavior of the signal. This differs from current popular methods of high frequency and envelope vibration analysis, where carrier demodulation is usually required prior to successful diagnosis using the fault frequencies [24].

Noise Analysis: We perform a CNN noise analysis using the same methodology described in Section IV-B.

Figure 5(b) shows the results of deep CNN configurations against various values of ν. The 2-layer CNN is slightly more resilient to corruption of the training data for some small level of ν, but at $\nu > 0.2$ the single layer CNN performs slightly better. The 3-layer CNN is unstable when varying ν. Regardless, the accuracy among all configurations remain relatively high, above 98% even for $\nu = 0.3$.

Figure 5(c) shows the results of varied number of layers of CNN that is trained with uncorrupted data, but is tested with corrupted data. It shows that when the input is corrupted up to $\eta = 0.10$, the deep CNN still has relatively high accuracy. At higher η, the deep CNN has a higher overall accuracy than the single layer CNN but the accuracy still similarly drops off steeply.

Finally, the effects of varying η and ν are tested against the deep CNN architecture. See Tables IV, V, and VI for results. Similar to the results found in IV-B, deeper architectures with ν below some threshold handle signal noise better.

V. Discussion

In the construction of the CNN models, we shunned the traditional signals processing methodologies and opted to directly input the normalized data into the CNN. For example, we found that as long as the input window size is large enough (e.g. more than 100 sample points), then the classifier accuracy doesn't improve even with larger window sizes. We also experimented with input windows with variable sizes of overlap, and found that the size of the overlap does not directly correlate to the accuracy of the final models. These findings imply that both the time taken and required computation resources to train CNN models can be minimized, because the size of the training data is reduced (since the size of window overlap can be zero), and that the input size can be restrained to lessen the memory footprint. The small number of filters required for a CNN to learn the data also reduces the training time. With our hardware setup (2.8 GHz Intel i7 QuadCore CPU, 16GB RAM), we are able to train most of the models under ten minutes. Thus, while neural network architectures in general are resource intensive, when applied to vibration analysis, it is a viable alternative to the manual signals processing techniques that may take days and weeks to complete.

		ν					
		0.05	0.1	0.15	0.2	0.25	0.3
η	0.05	0.9965	0.9960	0.9949	0.9903	0.9842	0.9702
	0.1	0.9903	0.9934	0.9902	0.9876	0.9831	0.9740
	0.15	0.9633	0.9851	0.9879	0.9797	0.9813	0.9689
	0.2	0.9074	0.9592	0.9737	0.9783	0.9778	0.9651
	0.25	0.7903	0.9079	0.9457	0.9640	0.9610	0.9626
	0.3	0.6969	0.8413	0.9007	0.9316	0.9465	0.9458

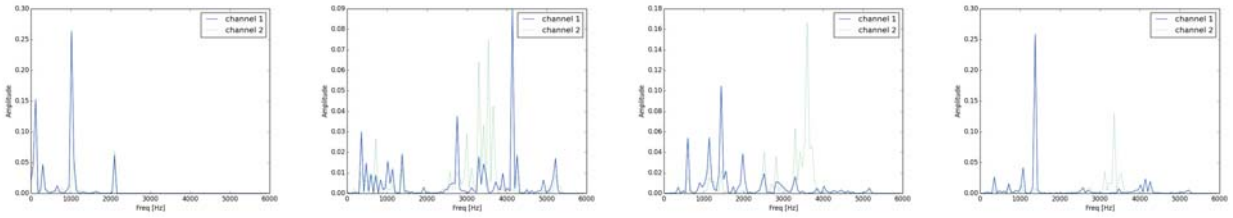
TABLE IV
Case Western Data Classification Accuracy, 1 Layer CNN, Corrupted Training and Test Set

		ν					
		0.05	0.1	0.15	0.2	0.25	0.3
η	0.05	0.9975	0.9965	0.9524	0.9907	0.9790	0.9691
	0.1	0.9881	0.9945	0.9803	0.9920	0.9813	0.9761
	0.15	0.9609	0.9865	0.9879	0.9907	0.9816	0.9679
	0.2	0.9052	0.9576	0.9721	0.9796	0.9739	0.9445
	0.25	0.8580	0.9121	0.9493	0.9626	0.9651	0.9649
	0.3	0.7276	0.8424	0.8898	0.9378	0.9521	0.9427

TABLE V
Case Western Data Classification Accuracy, 2 Layer CNN, Corrupted Training and Test Set

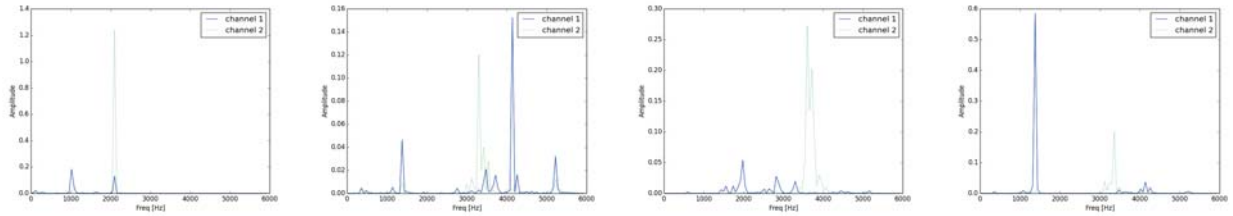
VI. Conclusion

In this paper, we have demonstrated that usage of CNN type architectures shows promise in raw vibration signal ball bearing analysis. The results, 98%-99% depending on the data set used, show that preprocessing of the input data to



(a) Raw Baseline (b) Raw Outer Fault (c) Raw Inner Fault (d) Raw Ball

Fig. 6. Case Western, Raw Input Data



(a) Smoothed Baseline (b) Smoothed Outer Fault (c) Smoothed Inner Fault (d) Smoothed Ball Fault

Fig. 7. Case Western, Smoothed Data

		ν					
		0.05	0.1	0.15	0.2	0.25	0.3
η	0.05	0.9972	0.9953	0.9943	0.9914	0.9868	0.9762
	0.1	0.9862	0.9869	0.9935	0.9916	0.9775	0.9810
	0.15	0.9721	0.9861	0.9844	0.9866	0.9851	0.9566
	0.2	0.9241	0.9621	0.9624	0.9375	0.9781	0.9614
	0.25	0.8585	0.9104	0.9470	0.9583	0.9375	0.9563
	0.3	0.7787	0.8650	0.9063	0.9391	0.9475	0.9550

TABLE VI
CASE WESTERN DATA CLASSIFICATION ACCURACY, 3 LAYER CNN, CORRUPTED TRAINING AND TEST SET

denoise or extract features is not required to achieve a high classification accuracy. The CNN is shown to be able to learn spectral frequency profiles for each category of fault with a very sparse architecture of 1-layer CNN and low amount of filters. It was found that the use of dual-channel data to train the CNN reduces the amount of filters necessary to achieve a high accuracy. Lastly, the addition of corrupted training input data and deep CNN architectures can each provide varying levels of resilience when there exists a certain threshold of signal noise.

REFERENCES

[1] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML '09. New York, NY, USA: ACM, 2009, pp. 609–616.

[2] H. Lee, P. Pham, Y. Largman, and A. Y. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," in *Advances in Neural Information Processing Systems 22*, Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta, Eds. Curran Associates, Inc., 2009, pp. 1096–1104.

[3] D. Palaz, M. Magimai.-Doss, and R. Collobert, "Analysis of cnn-based speech recognition system using raw speech as input," Idiap, Idiap-RR Idiap-RR-23-2015, 6 2015.

[4] W. Yan and L. Yu, "On accurate and reliable anomaly detection for gas turbine combustors: A deep learning approach," in *Annual Conference of The Prognostics and Health Management Society 2015*, vol. 6, 2015. [Online]. Available: http://www.phmsociety.org/node/1652/

[5] S. Tao, T. Zhang, J. Yang, and X. Wang, "Bearing fault diagnosis method based on stacked autoencoder and softmax regression," in *Control Conference (CCC), Hangzhou, China, July 28-30, 2015*. IEEE, July 2015, pp. 6331–6335.

[6] T. Junbo, L. Weining, A. Juneng, and W. Xueqian, "Fault diagnosis method study in roller bearing based on wavelet transform and stacked auto-encoder," in *Control and Decision Conference (CCDC), 2015 27th Chinese*. IEEE, 2015, pp. 4608–4613.

[7] http://www.mfpt.org/FaultData/FaultData.htm.

[8] http://csegroups.case.edu/bearingdatacenter/home.

[9] J. I. Taylor, *The Vibration Analysis Handbook*, 2nd ed. Vibration Consultants, 2003.

[10] S. J. Lacey, "An overview of bearing vibration analysis," 2007.

[11] E. Mendel, T. W. Rauber, F. M. Varejão, and R. J. Batista, "Rolling element bearing fault diagnosis in rotating machines of oil extraction rigs," in *17th European Signal Processing Conference, EUSIPCO 2009, Glasgow, Scotland, UK, August 24-28, 2009*, 2009, pp. 1602–1606.

[12] L. Tóth and T. Tóth, "On finding better wavelet basis for bearing fault detection," *Acta Polytechnica Hungarica*, vol. 10, no. 3, pp. 17–35, 2013.

[13] P. K. Kankar, S. Sharma, and S. P. Harsha, "Rolling element bearing fault diagnosis using wavelet transform," *Elsevier - Neurocomputing*, vol. 74, pp. 1638–1645, 2011.

[14] D. Scherer and S. Behnke, "Evaluation of pooling operations in convolutional architectures for object recognition," in *In: Artificial Neural Networks (ICANN), 20th Int. Conf*, 2010.

[15] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent, "Modeling Temporal Dependencies in High-Dimensional Sequences: Application to Polyphonic Music Generation and Transcription," in *Proceedings of the 29th International Conference on Machine Learning*. icml.cc / Omnipress, 2012, p. 244.

[16] A. rahman Mohamed, G. Dahl, and G. Hinton, "Deep belief networks for phone recognition," 2011.

[17] I. Goodfellow, H. Lee, Q. V. Le, A. Saxe, and A. Y. Ng, "Measuring invariances in deep networks," in *Advances in Neural Information Processing Systems 22*, Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta, Eds. Curran Associates, Inc., 2009, pp. 646–654. [Online]. Available: http://papers.nips.cc/paper/3790-measuring-invariances-in-deep-networks.pdf

[18] Y. LeCun, L. Bottou, G. B. Orr, and K.-R. Mueller, "Efficient backprop,"

in *In Neural NetworksTricks of the Trade, Lecture Notes in Computer Sciences 1524*. Springer, 1998, pp. 5–50.
[19] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, J. Frnkranz and T. Joachims, Eds. Omnipress, 2010, pp. 807–814. [Online]. Available: http://www.icml2010.org/papers/432.pdf
[20] Z. Chen, C. Li, and R.-V. Sanchez, "Gearbox fault identification and classification with convolutional neural networks," *Shock and Vibration*, vol. 2015, 2015.
[21] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P. antoine Manzagol, "Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion," 2010.
[22] Q. Xu, C. Zhang, and L. Zhang, "Denoising convolutional neural network," in *Information and Automation, 2015 IEEE International Conference on*. IEEE, 2015, pp. 1184–1187.
[23] H. Balderston, "Incipient failure detection: Incipient failure detection in ball bearings." DTIC Document, Tech. Rep., 1969.
[24] J. L. Frarey, "The history and application of the envelope detector," DTIC Document, Tech. Rep., 1996.

Bayesian Learning of Clique Tree Structure

Cetin Savkli , J. Ryan Carr, Philip Graff, Lauren Kennell
The Johns Hopkins Applied Physics Laboratory, Laurel, MD, USA

Abstract - *The problem of categorical data analysis in high dimensions is considered. A discussion of the fundamental difficulties of probability modeling is provided, and a solution to the derivation of high dimensional probability distributions based on Bayesian learning of clique tree decomposition is presented. The main contributions of this paper are an automated determination of the optimal clique tree structure for probability modeling, the resulting derived probability distribution, and a corresponding unified approach to clustering and anomaly detection based on the probability distribution.*

Keywords: Categorical, Probability, Anomaly, Clique Tree, Clustering

1 Introduction

With the rapid growth of categorical data available for analysis, the need for robust statistical approaches is becoming ever more critical. Unlike numerical data (such as weather or astronomical data), much of the data found in social networks, and the web in general, is categorical in nature. While methods for analysis of numerical data are well established, methods used for analysis of categorical data are more varied and still developing.

One of the challenges in the analysis of categorical data is a lack of a natural distance metric that most statistical learning algorithms rely on. This problem has led to numerous proposals and a comparative analysis of alternatives can be found in [1]. While the lack of a natural distance metric is a problem, it is also known that as the dimensionality of the attribute space increases the distance metrics become less and less useful, a fact that is also known as the "curse of dimensionality." In particular, the curse of dimensionality implies that in high dimensions the data becomes sparse (thinly spread over the total event space) and thus most of the data becomes equally anomalous. Categorical data, such as cyber or financial transactions, is often high dimensional and can easily comprise dozens of attributes. Therefore, identifying anomalies becomes a challenging task in many categorical data sets. Rule based or ground truth based classification approaches can be used to detect predefined event classes, but these methods have limited utility to detect new anomalies, meaning the rare events which have not been previously characterized. However, the anomalies of greatest interest may be new anomalies, and in fact they may be one-time events which do not form a cohesive class on which to train a classifier.

The inability to reliably identify anomalies has practical consequences as human inspection of large data sets for anomalies is a time-consuming activity and impractical on large data sets. Therefore, it is desirable to develop robust analytic approaches that do not require ground truth, do not rely on a distance metric, and that can handle the high dimensionality of the categorical data.

In this work we explore a probabilistic approach to data representation that addresses the challenges described above. The approach is based on constructing a joint probability distribution using a structure called a clique tree (also known as a junction tree). The clique tree expresses dependencies in a high dimensional attribute space and can be used to make probabilistic inferences about data ([2], [3]). The clique tree approach is analogous to density estimation for numerical domains, but is more general as it can be used to infer probabilities of both numerical and categorical data.

Clique tree structures are an active area of research. Research about clique trees in literature has generally focused on inference algorithms as well as building of clique trees with a narrow width for fast inference. For instance, searching for a clique tree that satisfies various compactness criteria is discussed in [4], and clique tree based inference is discussed in [5] and [6].

The focus in this paper is the determination of an optimal clique tree structure which best represents attribute dependencies within the data and thus the information content of the data. The main contributions of this paper are:

- An automated and parameter-free determination of an optimal clique tree structure for probability modeling using Bayesian learning from data;
- A unified approach to clustering and anomaly detection based on the derived probability distribution.

In the following sections, we present the construction of a probability distribution, specification of the optimal clique tree structure directly from data, and applications to anomaly

detection and clustering. Using a publicly available categorical data set, we present results which show that Bayesian learning can be used to construct an optimal clique tree that maximizes the probability of observed data while providing inference capability for unseen data.

2 The joint probability distribution

In high dimensional data sets, there is generally insufficient data from which to characterize probabilities; the available data points are spread too thinly over a very large space of possible attribute combinations. However, if it is known that some subsets of the variables are independent from other subsets of variables, a joint probability distribution in a high dimensional space can be decomposed into a product of lower dimensional probabilities ([2], [3]). Within low dimensional spaces, the data is more concentrated and a probability distribution can be successfully derived. Therefore, as the first step in deriving a joint probability distribution, dependencies will be characterized using the mutual information. The choice of mutual information as a metric for correlation is motivated by the generality of this metric in handling categorical and numerical data. However, it is possible to use another correlation metric as the approach is independent of a chosen metric. From the mutual information, an optimal structure for clique trees is obtained, and the probability model is based on this optimal clique tree structure.

2.1 Mutual information graph

For two variables X and Y, the mutual information $I(X,Y)$ is defined by:

$$I(X,Y)=\sum_{x\in X}\sum_{y\in Y}p(x,y)\log\left(\frac{p(x,y)}{p(x)p(y)}\right) \quad (1)$$

A normalized version of the mutual information [7] can be used to establish the degree of dependence between attributes. The normalized mutual information (NMI) is given by

$$M(X,Y)=\frac{I(X,Y)}{\min(H(X),H(Y))} \quad (2)$$

where variable entropy H is defined as

$$H(X)=-\sum_{x\in X}p(x)\log(p(x)) \quad (3)$$

The NMI varies varies between 0 and 1, where 0 indicates independence while 1 implies complete dependence.

To illustrate the ideas in this paper, a data set of mushrooms will be used [8]. The data set contains 8,124 mushrooms, each characterized by 22 attributes including color, shape, odor, etc., which are denoted in this paper by "a1" through "a22". For each pair of attributes, the NMI is computed, and the distribution is shown in Fig. 1. The distribution indicates that there are a small group of attributes with strong dependence while most attribute pairs have weaker dependence.

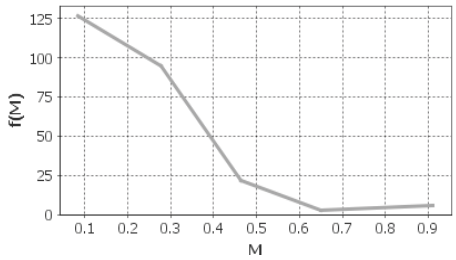


Fig. 1. Number of attribute pairs (f(M)) with a given mutual information (M) in the mushroom data set.

The mutual information results can be represented as a weighted graph. The 22 attributes are the graph nodes, and the link between each pair of nodes is weighted with the NMI of that attribute pair. Links can then be pruned with a threshold, so that only the links that indicate strong dependency are retained. Specification of this threshold is a key focus of this paper and is discussed in detail in Section 2.3.

2.2 Clique tree and joint probability model

For ease of illustration, we continue the derivation of a clique tree and probability model using a hypothetical data set with only six attributes $\{a, b, c, d, e, f\}$. The original NMI graph on these attributes would contain 15 weighted links. Suppose a threshold is chosen so that only the links in Fig. 2 are retained.

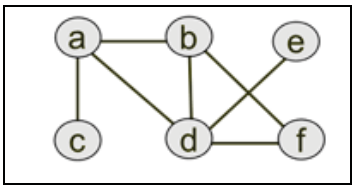


Fig. 2. Example of a pruned mutual information graph.

The pruned mutual information graph provides a basis for constructing a clique graph and clique tree. The first step is to find chordless cycles in the pruned mutual information graph and fix them. This is a necessary condition for the clique tree to satisfy the running intersection property, which guarantees that the clique tree will provide a joint probability distribution that is normalized [9]. A chordless cycle is a cycle such that nodes on the periphery have no direct connection to each other except for the nodes which are adjacent in the cycle. The pentagon-shaped cycle in Fig. 3 is an example of a chordless cycle. Fixing the chordless cycle can be accomplished by introducing the links shown on the right side of the figure.

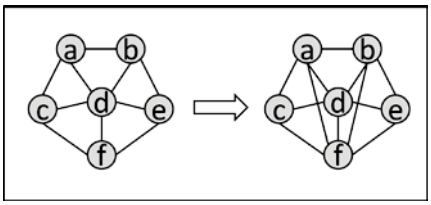


Fig. 3. On the left, the pentagon outline is a chordless cycle. This is fixed by adding the two additional edges shown at right.

After chordless cycles are repaired in the mutual information graph, we construct the clique graph. To form a clique graph, the maximal cliques of the input graph become the nodes of the clique graph. For instance, in the graph in Fig. 2, the node set $\{a,b,d\}$ is a maximal clique, and therefore $\{a,b,d\}$ becomes a node in the clique graph. Two clique graph nodes are linked if the cliques have at least one underlying node in common. We also label the link by those overlapping nodes, which are called the separator set. Lastly, to construct the clique tree from the clique graph, we use the minimum spanning tree algorithm where the link distances are measured in inverse of the separator set size [9]. The right side of Fig. 4 is the clique tree for the graph from Fig. 2. The ovals represent the maximal cliques, and the rectangles are the separator sets on the links.

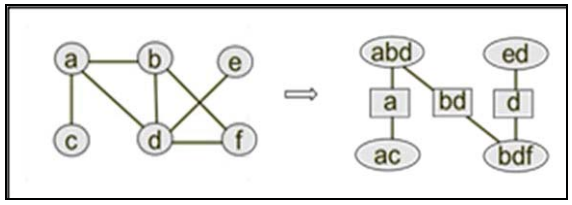


Fig. 4. The mutual information graph from Fig. 2 and its clique tree representation.

The derived probability model is easily read from the clique tree. For any data point (a,b,c,d,e,f), its probability is given by the product of the probabilities within subspaces corresponding to maximal cliques, divided by the probabilities of the subspaces given by the separator sets:

$$P(a,b,c,d,e,f) = \frac{P(a,c)P(a,b,d)P(b,d,f)P(e,d)}{P(a)P(b,d)P(d)} \qquad (4)$$

Some of the benefits of this probability decomposition are immediately clear from its structure. First, the right side of equation (4) contains only lower dimensionality probabilities. The lower dimensional probabilities can be more reasonably inferred from available data, which addresses the problem of data sparsity and the curse of dimensionality. Second, the subspaces in the numerator of equation (4) consist of highly dependent attributes. When attributes are dependent, unexpected combinations of data values will immediately stand out; they will violate the expected behavior encoded in the attribute dependency. For instance, an unusual combination of values for (b,d,f) would stand out as an anomaly in that subspace. The factor $P(b,d,f)$ will have a relatively low value, and in turn this will tend to cause $P(a,b,c,d,e,f)$ to have a low value as well. Therefore, the probability model correctly finds anomalies because it judges a data point by its conformance to expected variable dependencies. We will return to this idea in Section 3. In Section 2.3, we focus on determining the optimal mutual information threshold, and its relationship to the generalization capability of the distribution.

2.3 Generalization capability of the distribution

An immediate question regarding the joint probability distribution is related to generalization capability. The problem of generalization is a familiar one from other areas of data analysis. For example, fitting data to a curve presents a trade-off between the accuracy of the fit and how well it explains the data points not yet measured. In the case of clique tree decomposition the problem is related to the extent of the pruning of the mutual information graph that forms the basis of the clique tree. Consider the two extreme cases:

Case 1: If the NMI threshold is set to 0, all of the mutual information links are retained. The resulting probability distribution is the full joint distribution. The probability of any data point becomes equal to how frequently that data point is observed, and any data point not previously seen will be considered to have probability of zero. The left figure in Fig. 5 shows this type of distribution. In short, this threshold is an overfitting of the data. This is again why high dimensionality and data sparsity is clearly problematic for modeling a probability distribution, since there are presumably many combinations of attributes which are not inherently anomalous but are assigned a zero probability.

Case 2: If the NMI threshold is set to 1, all of the mutual information links are removed. This limit corresponds to fitting the data assuming that all dimensions are independent (also known as Naïve Bayes assumption). Treating attributes as independent leads to a distribution that covers the entire parameter space and corresponds to an overly smooth distribution. According to this distribution, shown on the right side of Fig. 5, all data has small and non-zero probability.

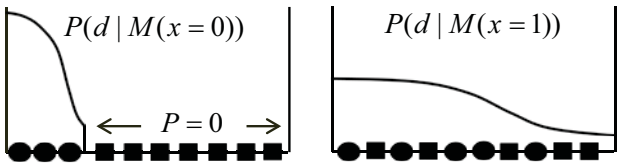


Fig. 5. Left: Setting mutual information threshold $x = 0$ results in unobserved data (squares) having zero probability. The average probability (P) of observed data (circles) is maximized, and entropy (S) is minimized. Right: Setting $x = 1$ results in all data points (observed and unobserved) being assigned positive probability, which maximizes the entropy.

Neither of the extreme cases is satisfactory. A threshold is needed which balances the competing goals of fidelity to observed data, and allowance for previously unseen data. In order to choose an optimal threshold x, we seek to maximize $P(M(x)|D)$ where D is the set of observed data and $M(x)$ is the probability model derived from the clique tree corresponding to threshold x. While each threshold leads to a specific NMI graph, the relationship is not reversible. Threshold x lives in a continuum while there are a finite set of trees that can be constructed using these thresholds. Using Bayes' Theorem, the posterior distribution for the model $M(x)$ specified by an NMI threshold x can be expressed as

$$P(M(x) \mid D) = \frac{P(D \mid M(x))P(M(x))}{P(D)} \qquad (5)$$

The prior $P(M(x))$ can be assumed to be uniform, and $P(D)$ is a normalization factor independent of x. Therefore, the problem is equivalent to maximizing $P(D|M(x))$, which is the product of individual data probabilities. However, even though the probability of the observed data needs to be maximized, using all available data for this purpose leads to over-fitting of the distribution leading to an optimal threshold of $x = 0$. It is necessary to force the distribution to assign mass to a more expansive set of data points than those it trains on, but not across the entire attribute space.

The solution is to divide the observed data into a training set and a test set. With this partition, the quantity we wish to maximize is given by

$$P(D \mid M(x)) = P(D_{Train} \mid M(x))P(D_{Test} \mid M(x)) \qquad (6)$$

If any of the test data is assigned zero probability, the right side of equation (6) becomes zero, and the corresponding x will therefore not be chosen as optimal. The idea is shown pictorially on the left side of Fig. 6.

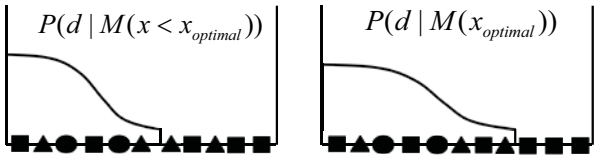


Fig. 6. Circles represent training data, triangles represent test data, and squares represent unobserved data. Left diagram: If the threshold is too low, some of the test data will have probability of zero. This threshold is rejected by maximizing $P(D|M(x))$ in equation (7). Right diagram: The threshold is just large enough to assign positive probability to all of the training and test data.

When the NMI threshold is low, the distribution will accommodate the training data, but it will be too compact to explain the test data. At the optimal threshold, shown in the right side of Fig. 6, the distribution assigns positive probability to the training data and the test data, and probably also to some of the possible data points which are still unseen. If the threshold were higher than optimal, $P(D|M(x))$ would start to decrease, since more of the mass would be assigned to the unobserved data. In summary, the correct value of x explains the training data, and accommodates the test data (so that plausible unseen data is allowed), but it does not spread the distribution needlessly wide over the total space of attribute combinations.

For the mushroom data set, the data is randomly divided into training and test sets, with 80% of the data assigned to the training set. Note the average probability of observed data decreases as x increases (Fig. 7) since more of the mass is shifted onto unobserved data points.

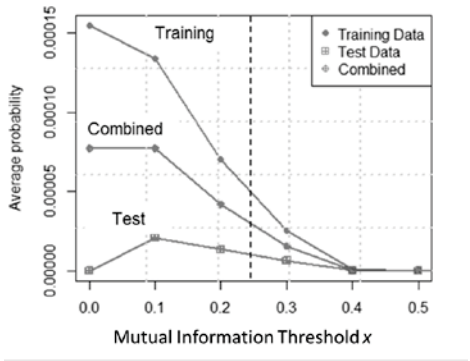


Fig. 7. Average probability of training data, test data, and all observed data combined, as a function of NMI threshold.

Maximizing $P(D|M(x))$, the quantity in equation (6), the optimal threshold for the mushroom data is found to be $x = 0.243$. The plot of $\log(P(D|M(x)))$ is shown in Fig. 8. For $x < 0.243$, $P(D|M(x)) = 0$ since at least some of the test data is not accounted for; this is the idea of the left side of Fig. 6. As x increases past 0.243, $P(D|M(x))$ rapidly decreases monotonically. Note it is also possible to repeat the analysis for multiple partitionings of the data into test and training sets to improve robustness of the threshold determination.

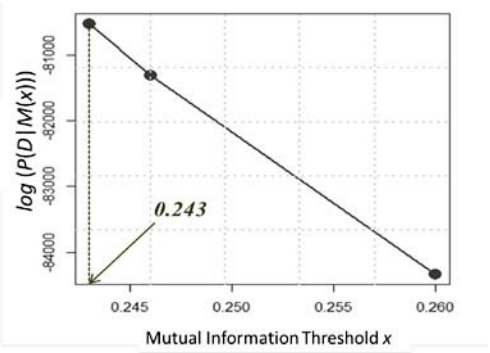


Fig. 8. Plot of $\log(P(D|M(x)))$ as a function of NMI threshold.

An interesting property of this solution is related to the entropy of the clique tree. Entropy is a function of the model $M(x)$, since the threshold determines clique tree structure. It can be calculated in terms of vertex and edge clique entropies:

$$H(M(x)) = \sum_{i \in V} H(C_i) - \sum_{ij \in E} H(C_{ij}) \quad (7)$$

where C_i is a clique vertex in the clique tree and C_{ij} is an edge clique (separator set). This expression reduces the calculation of entropy for a high dimensional distribution to the calculation of individual clique entropies which are simpler to calculate. As expected, the entropy of the clique tree probability distribution increases as the threshold x increases (Fig. 9). In this plot the dashed line marks the location of the optimal NMI threshold where the feasible region for the threshold is to the right of this boundary. As this result indicates, the optimal threshold that maximizes the posterior probability corresponds to minimum entropy solution in the domain where the test data has non-zero probability.

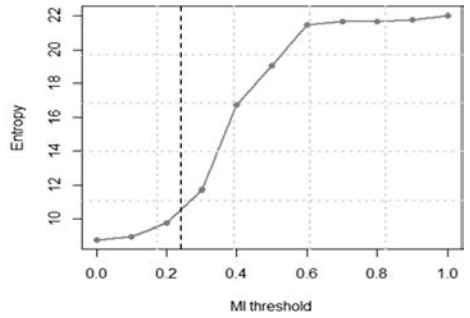


Fig. 9. Entropy of the clique tree probability distribution as a function of mutual information (MI) threshold. The dotted line shows the location of the optimal threshold 0.243.

Using the optimal threshold of 0.243, the corresponding optimal clique tree structure is shown in Fig. 10. Each node represents a clique, and is labeled with its attributes. The separator sets (the overlapping attributes between cliques) are not labeled on the graph; however, the cliques are linked if they have at least one attribute in common, and the separator sets are easily inferred by examining the node labels. Note that a few attributes are independent of all the others, but the typical clique size is in the range of 6-10 attributes.

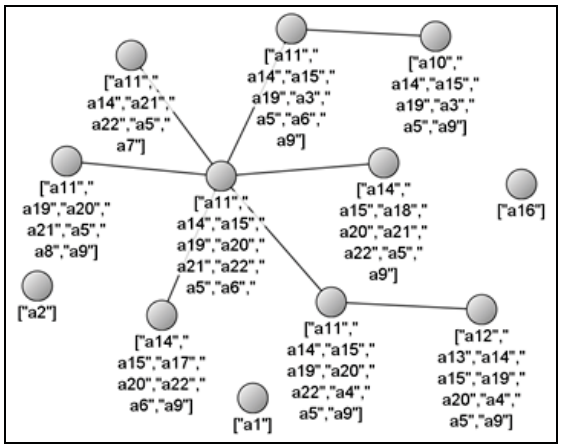


Fig. 10. Optimal clique tree structure for the mushroom data set.

3 Clustering and anomaly detection

Using the clique tree decomposition, we can examine data from the perspective of clique-based clustering, and discuss how cliques provide insight into anomaly detection and characterization.

3.1 Clique-based clustering

Clustering categorical data is challenging since many clustering approaches rely on distance metrics, and are therefore inapplicable to categorical data. For categorical data, some clustering approaches aim to find one optimal clustering ([10],[11],[12]). By contrast, the clique tree structure promotes alternative clusterings based on subspaces defined by the cliques of the strongly coupled attributes.

Clique-based clusterings provide a natural indexing mechanism for data where each clique provides an index and each data point is described by the particular combination of attributes that belong to that clique. For example consider a clique $C_{14,17,6}$ formed by strongly coupled attributes (14,17,6) whose value multiplicities are (9,3,3). Based on the multiplicities of the variables involved there are 81 possible value combinations in the clique subspace. However, when the mushrooms are clustered according to their attribute combinations, only 11 clusters are found, with membership distribution shown in Fig. 11. Furthermore, the sizes of these clusters are highly skewed in that only 5 of the 11 clusters have a significant population.

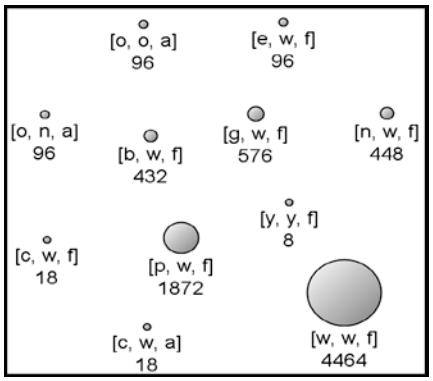


Fig. 11. Cluster memberships for a clique formed by attributes 14, 17, and 6. Each cluster is labeled by the ordered triplet of data values for the three attributes, as well as the membership count. Of the 81 possible attribute value combinations in this attribute subspace, only 11 value combinations were observed in the data.

For the clique structure in Fig 10, each mushroom would have potentially 12 indices defined by the 12 vertex cliques. The indices can be used to infer links between entities to form an association graph for data points. For example, a similarity measure between data points can be defined by measuring how often two data points fall into same clusters (share the same indices) according to different clique clusterings. The use of clique indexing for clustering categorical data is a

promising direction for future exploration as it does not require a distance metric in categorical space.

3.2 Anomaly detection

One aspect of the "big data problem" is the challenge of finding events of interest, such as suspicious events, from very large data sets. Rule-based and classification-based approaches can assist this process by labeling some types of events, namely those with attribute combinations corresponding to a previously identified type of behavior. Anomaly detection is a more general approach to finding events of interest. In this approach, there is no ground truth to work from. In fact, part of the motivation for taking this approach is to discover anomalous or suspicious events which have not been previously characterized as such.

The intuition behind the anomaly detection approach is that most people or entities are engaged in innocent (or uninteresting) behavior most of the time. Put another way, suspicious events should be unusual events. A few remarks are necessary about this assertion: First, some suspicious or malicious events may actually be relatively common; however, we assume that commonplace events are detectable by other means, and it is not the goal of an anomaly detector to discover them. Second, unusual events may encompass many innocuous events as well as suspicious ones; a variety of one-time or idiosyncratic behaviors may be observed which are perfectly innocent. Even with these caveats, the important point is that suspicious events should be concentrated toward the low tail of the probability distribution.

The clique-based probability distribution derived in Section 2 assigns a probability value to all possible attribute value combinations, and thus can be used to find anomalous events. Further, the clique decomposition such as that in equation (4) also provides insight into why a data point is anomalous, by examining the probability values of each of the factors that comprise the right hand side of the probability expression. This insight may assist in further investigation of the event. Returning to the example of the mushroom data and using Fig. 11, the distribution for clique $C_{14,17,6}$ shows that most of the mushrooms belong to 5 clusters, whereas 6 of the clusters have low membership. Attribute values corresponding to these clusters are possibly indicative of anomalous mushrooms.

In comparing clique probability values to elucidate the reason for an anomaly, it should be noted that cliques comprising more attributes will exhibit lower probability values in general. This is because there are more possible attribute value combinations and thus the data is distributed across a larger attribute space. Therefore, to perform clique comparisons, the clique probabilities should be normalized. One way to achieve this is to derive percentiles (or cumulative distribution functions) from the probability distributions for each clique.

Taking the investigative process one step further, when a clique is identified whose value combination was the reason for the anomaly, the analyst may query for other events with the same attribute value combination. This returns the cluster of similar events according to the ideas of Section 3.1. These other events may or may not be considered anomalies; their overall probabilities also depend on probability values in other cliques. Whether the similar events are anomalous or not, viewing the events as a group based on the clique clustering may be informative in characterizing behavior patterns.

4 Related work

Anomaly detection is one of the fundamental challenges in statistical inference with a rich literature. A recent survey of this field is provided in [13]. One of the difficulties in anomaly detection is the fact that training of an anomaly detector has to be performed without a reliance on ground truth as anomalies by definition do not form a cohesive class. Therefore, there is no single right answer for what constitutes an anomaly, and performance of anomaly detection is therefore harder to adjudicate than classification which can be trained from labeled classes. Approaches to anomaly detection in categorical attribute spaces must also overcome the lack of a distance metric. An alternative parameter free anomaly detection approach based on data compression is found in [14] and [15]. Work in these references relies on minimum description length and does not require a distance metric.

The method presented in this paper falls into the category of likelihood-based approaches. There are similarities between the approach presented and the method proposed in [16], in that probabilities are estimated using decomposition into lower dimensional spaces using groups of related attributes. There are also similarities to Bayesian network based anomaly detection proposed in [17] where structure of the network is determined by domain expert input. In comparison the method we propose does not require any human input to specify the structure of the probability distribution. The main distinctions are that our approach optimizes the (often overlapping) subsets of attributes in the probability decomposition using a clique tree structure. Although we propose a method to automate the derivation of the optimal threshold and thus the probability distribution, the user may still choose to shift the threshold, or may combine the results of using different thresholds by using Bayesian model averaging. In this way the user's domain knowledge is allowed to influence the probability estimations. Using attribute subspaces for analysis has some similarity to another approach [18] that is based on subspace based anomaly detection. A key difference of the presented approach is that it is a probabilistic model that can be used for any combination of categorical and numerical data.

5 Implementation

While computational aspects of the problem are not the main focus of this paper, the probability modeling, anomaly detection, and classification methods described in this paper were implemented using a scalable platform, Socrates [19]. All steps of the computations including computing the mutual information and clique tree optimization are parallelized in Socrates to accommodate large data sets.

6 Conclusions

In this paper a clique tree approach to categorical data analysis has been presented with particular focus on the problem of learning of the clique tree structure for probability modeling and anomaly detection. The clique tree approach produces a probability model which exploits variable dependencies to decompose the joint probability into a product of joint probabilities in lower dimensions. By using lower dimensional subspaces, the probability model overcomes the problem of data sparsity, or the curse of dimensionality. At the same time, the probability decomposition provides clear anomaly signatures since it judges the likelihood of a data point by its conformance to expected variable dependencies in the lower dimensional subspaces. Finally, it has been shown that it is possible to use a Bayesian approach to determine a threshold that specifies the optimal structure of the clique tree. The optimal clique tree structure results in the probability model which best balances the competing requirements of fidelity to observed data and accommodating previously unseen data values.

7 References

[1] S. Boriah, V. Chandola, and V. Kumar, "Similarity measures for categorical data: A comparative evaluation," SDM 2008: 243-254.

[2] C. Chow and C. Liu, "Approximating discrete probability distributions with dependence trees," IEEE Transactions on Information Theory, 1968.

[3] N. Robertson and P. S. Seymour, "Graph minors," J. of Combinatorial Theory, Series B 36: 49–64, 1984.

[4] J. Pearl, "A distributed hierarchical approach". Proc. of the 2nd National Conf. on AI. (AAAI-82), AAAI Press., pp. 133–136 (1982).

[5] J. Pearl, "Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference," (2nd ed.). San Francisco, CA: Morgan Kaufmann, 1988.

[6] D. Shahaf, A. Chechetka, and C. Guestrin, "Learning thin junction trees," Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS), Clearwater Beach, FL, USA. Vol. 5 of JMLR: W&CP 5, 2009.

[7] Y. Yao, "Information-theoretic measures for knowledge discovery and data mining," in Entropy Measures, Maximum Entropy Principle and Emerging Applications, Karmeshu (ed.), Springer, pp. 115–136, 2003.

[8] Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

[9] D. Koller and N. Friedman, "Probabilistic Graphical Models," MIT Press., pp. 1208., ISBN 0-262-01319-3, 2009.

[10] V. Ganti, J. Gehrke, and R. Ramakrishnan, "CACTUS: Clustering categorical data using summaries," Proceedings of the ACM-SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, 1999.

[11] S. Guha, R. Rastogi, and K. Shim, "A robust clustering algorithm for categorical attributes," Information Systems, pp. 512-521, 1999.

[12] M.X. Vinh and M. Houle, "A set correlation model for partitional clustering," Advances in Knowledge Discovery and Data Mining, 14th Pacific-Asia Conference, 2010.

[13] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey" ACM Computing Surveys (CSUR), Surveys Homepage Archive, Vol. 41, Issue 3, July 2009.

[14] L. Akoglu, H. Tong, J. Vreeken, and C. Faloutsos, "Fast and reliable anomaly detection in categorical data," Proceedings of the 21st ACM International Conference on Information and Knowledge Management, 2012.

[15] K. Smets and J. Vreeken. "The odd one out: Identifying and characterising anomalies," In SDM, SIAM, 2011.

[16] K. Das and J. Schneider, "Detecting anomalous records in categorical datasets," Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2007.

[17] A. Bronstein, J. Das, M. Duro, R. Friedrich, G. Kleyner, M. Mueller, S. Singhal, and I. Cohen. Using bayes-nets for detecting anomalies in Internet services. In INM, 2001.

[18] C. Aggrawal and P. Yu, "Outlier detection for high dimensional data," Proceedings of the ACM SIGMOD International Conference on Management of Data, 2001.

[19] C. Savkli, R. Carr, M. Chapman, B. Chee, and D. Minch, "Socrates: A system for scalable graph analytics," Proceedings of the IEEE High Performance Extreme Computing Conference, 2014.

Mixtures of Polynomials for Regression Problems

J. Carlos Luengo, and Rafael Rumi

Abstract— **This paper presents a novel methodology for solving regression problems, based on Bayesian networks. Graphical models have been used mostly for solving classification problem, but they can be extended to solve regression problems by allowing continuous variables in the model. Gaussian models were first proposed for dealing with continuous variables in Bayesian networks, but they have some limitations. The Mixtures of Polynomials have recently been proposed as a valid alternative to the Gaussian model, but it has not yet been tested in regression models. For this, a model is developed within a restricted Bayesian network (Naive Bayes), and the parameters involved analyzed. Two preprocessing steps are presented to improve the results, and the procedures for learning the corresponding distributions shown. The results are compared with some state-of-the-art models, obtaining promising results.**

I. Introduction and related work

Regression problems are one of the main problems in data mining. They are present in all science fields, dealing with many real-world problems, and therefore they have attracted much attention from the statistics and data mining community. A regression problem can be seen as a supervised classification problem, in which the goal variable is continuous or numeric. There is a great number of methods to solve regression problems; one of them are graphical models, and Bayesian networks in particular, which have recently raised as valid alternatives for classification problems.

Bayesian networks are known to be efficient tools for probabilistic reasoning, that can be applied also to classification problems, both supervised and unsupervised, due to their flexibility. In the case of supervised classification, if the goal variable is continuous, we are facing a regression problem, that most times is solved by a fixed structure model, in such a way that the number of parameters to estimate is minimized.

In order to incorporate continuous variables in these regression models, a structure compatible with discrete and continuous variables is presented, the Mixtures of Polynomials (MoPs), which have been recently developed in Bayesian networks [1], but not applied to regression problems.

The main goal of this paper is to show that MoPs models are a competitive model for regression problems, in comparison to well known state-of-the-art methods, and to analyze the impact on the results of the different parameters of the model.

A Bayesian network is a decomposition of a joint probability distribution in products of conditional distributions. It can be expressed by a directed acyclic graph, in which every node in the graph represents a random variable in the problem, and the presence of an edge linking two nodes represents a statistical relation between them. Associated to each node in the graph there is a probability distribution of the corresponding variable, given its parents in the graph. In the case of root nodes (without parents in the graph) this distribution reduces to a marginal distribution.

J. Carlos Luengo is with the Alyanub IES, Department of Mathematics, Almeria, Calle Mayor, 58, 04630, Vera, Almeria, Spain (email: jcluengolopez@gmail.com).

Rafael Rumi is with the Department of Mathematics, Almeria University, 04120, Almeria, Spain (email:rrumi@ual.es).

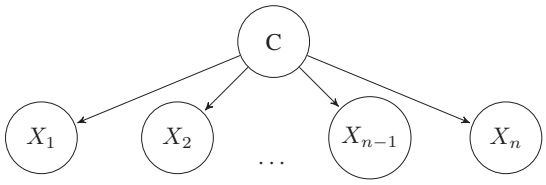


Fig. 1. Structure of Naive Bayes classifier, in which C is the goal variable, and the rest are the feature variables.

Let $X_1, X_2, \ldots, X_n$ be the random variables defining our problem, then the joint probability distribution of the variables reduces then to the following decomposition, because of the independence relation encoded in the network [2]:

$$p(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} p(x_i|pa(x_i))$$

where $p(x_i)$ represents the parents (variables) of X_i in the graph.

When the main purpose of the model is to predict a goal variable, *i. e.* a classification problem, it is usual to employ a restricted Bayesian network, where the restriction is placed on the structure of the network. Examples of these models are kDB [3], Tree Augmented Network (TAN) [4] , Forest Augmented Network (FAN) [5], and the simplest model, the Naive Bayes (NB) model, in which all the features are assumed to be independent given the class. In the NB model, the class variable is the only root variable, which is linked to every feature variable in the model. An example of a NB model can be seen in Fig 1.

As simple as it may seem, NB is known to be an accurate classifier with a relatively small computational complexity. In this kind of models, the focus is placed on maximizing the accuracy of the prediction, rather than in estimating accurately the model parameters [6]. The rest of the restricted-models mentioned before increase the complexity of the model, relaxing the independence assumption of the NB, and so including more links in the graph.

A. Continuous variables

Bayesian network models were developed originally for discrete, *i. e.* categorical, variables; however in most of real-

world applications there exist continuous data. Practitioners and researchers usually solve this issue by discretizing the continuous variables and transforming them into discrete, by means of some common techniques, such as *equal frequency*, *equal width*, or *k-means* [7], or even some context-specific methods such as dynamic discretization [8] or classification-oriented methods [9]. All these solutions conveys a loss in information, and should be avoided when dealing with a regression problem, in which the goal variable is continuous, and so discretizing would intrinsically change the nature of the problem.

Therefore, methods able to include continuous variables in BNs are needed, and so, the Conditional Gaussian Model (CLG) [10] was conceived to overcome this problem, by defining the continuous variables as Gaussians. Even though this model is still widely used, it has two main disadvantages, the distribution of the continuous variables must be Gaussian, and discrete nodes cannot have continuous variables as parents. In particular, in the case that we use a NB model for regression, if any of the feature variables is discrete, the CLG model cannot be used because of this second constraint.

Following this idea appears the Mixtures of Truncated Exponentials model (MTE) [11], in which any probability distribution can be approximated and there is no restriction in the topology of the network. This model has been successfully applied in recent years to classification and regression problems [12], [13] and in general to hybrid BNs [14], [15]. The Mixtures of Polynomials model [1], [16] is similar, in the sense that it can also approximate any probability distribution, but uses polynomials as the basis functions, gaining fitting power just increasing the degree.

Therefore, in this paper we propose the use of MoPs models within Bayesian networks to deal with regression problems, and compare it with state-of-the-art methods. Section II-B presents the details of the MoP model used in this paper to represent the continuous variables.

There is not much literature on regression using mixed BNs. Previous works containing some connections with this current paper deal only with estimating MoPs distributions or using MTEs for regression problems.

MoPs have mostly been applied to approximate specific known distributions, as in [19], but not to include them as a tool to approximate any dataset, neither to use them for regression. Therefore, the parameter estimation procedure is based on minimizing the Mean Squared Error (MSE) between the origina data and the fitted distribution, rather than improving accuracy of forecasts. There are some recent publications that deal with this problem using different properties of the polynomials [22], [25], based on B-splines approximations and maximum likelihood, instead of focusing in accuracy indicators. MoPs have also recently been shown [25], [18] to be a valid alternative when facing a classification problem, using both a NB and a TAN structure, however they have not been applied yet to regression problems, even though the results obtained for classification problems were satisfactory.

MTEs have been used to solve regression problems following a similar general structure to this paper, but maintaining strong differences in terms of estimation of the parameters of the model, based only on least squares approaches (for more information, see [13]).

The rest of the paper is structured as follows: Section II presents the details of the methodology proposed, in Section III an exhaustive set of experiments are carried out, and the paper ends with some conclusions.

II. Proposed Methodologies

A. Preprocessing steps

Before carrying out the main procedure of learning the model, some preprocessing task are executed, in order to simplify or enhance the results.

1) Feature selection: Selecting the appropriate variables in a classification model is a crucial step for successfully solve the problem, since it avoids overfitting and noise in the model [17]. We have selected 3 *filter-wrapper* strategies, based on the mutual information between the goal variable and each feature. Although this procedure is included as a preprocessing step, it is actually embedded into the learning algorithm, not a as different step.

The computation of the mutual information is expressed in the following formula

$$MI(X;C) = \sum_{i=1}^{n}\sum_{j=1}^{m} p(x_i, c_j) log \frac{p(x_i, c_j)}{p(x_i)p(c_j)}$$

obtained from the discretized database. We have chosen to discretize due to the ease and speed of this procedure, because this is just an initial step, that takes into account the information shared between the two variables, and is later updated in terms of accuracy of the prediction, which is more suitable to the current problem.

Once the mutual information has been obtained, the variables are ranked, and they are included (or not) in the model, according to three different heuristics:

- Type 1: Classical filter-wraper approach: Variables in the ranking are inserted in the model, as long as they improve the results, by decreasing the MSE (see Section III). The first time a variable does not improve the results, we stop.
- Type 2: Variables are ranked, and the three first variables not yet inserted in the model are considered. We create new model inserting the first one, if the error decreases, it is finally inserted. If not, we try with the second one. If it decreases the error, it is finally inserted. If not, we try again with the third one. Only if none of them improves the result, we stop. Once a variable is included, a new subset of the three best-ranked variables is considered.
- Type 3: Equivalent to Type 2, but we allow some increase of the MSE, only three times. The first one we allow a 10% increase, the second one a 8% decrease, and the third one a 5% decrease. This method resembles to the *Simulated Annealing*.

2) Discretization: Detecting when a variable is discrete or continuous is easy when looking at the definition, however, when dealing with real data, this difference is not straight. Some variables cannot be considered discrete neither continuous because of the great number of levles, or the difficulty for fitting a polynomial. Therefore, we have considered a special type of variable, *pseudo-continuous* variables, defined in [18], found mainly in large datasets, to treat those variables continuous by definition, but not in practice.

These *Pseudo-continuous* variables are defined as numeric features which have more then 20 different values, but less than the 5% of the total number of observations of the dataset.

In the *Experiments Results* section we will investigate if detecting and discretizing these variables yields in an accuracy-gain, as it was observed for classification problems in [18].

B. Learning the model

Learning a NB model from data using MoPs to represent the continuous variables reduces to estimating the corresponding parameters. According to Fig. 1, these are : 1) a marginal distribution for the goal variable, $f(c)$ 2) a conditional distribution for each feature variable, given the goal variable, $f(x_i|c)$. These distributions are represented within the BN by means of the MoP model.

1) Mixtures of Polynomials: The MoP framework is able to approximate any continuous distribution by a piecewise function which has in each piece a polynomial function. This allows to work directly with the continuous variables without the need of discretizing them. A MoP function is defined as follows [1]:

Definition 1: A one-dimensional function $f : \mathbb{R} \rightarrow \mathbb{R}$ is said to be a mixture of polynomials function if it is a piecewise function of the form

$$f(x) = \begin{cases} a_{0i} + a_{1i}x + \cdots + a_{ni}x^n & x \in A_i \quad , i = 1, \ldots, k \\ 0 & \text{otherwise} \end{cases}$$

where $A_1, \ldots, A_k$ are disjoint intervals in $\mathbb{R}$ and $a_{0i}, \ldots, a_{ni}$ are real constants for all i.

There are some variants to this definition, in [16] Shenoy *et al* re-defines the MoP function to include multivariate functions in which $A_1, \ldots, A_k$ may be also hyper-rhombuses. This re-definition makes MoP functions richer because now they can deal with deterministic functions, however it makes computations extremely complex [19], [20]. We do not use this variant because, in the NB model for regression, we need a tradeoff between accuracy and complexity.

The main reason for using MoPs is that they have a great fitting power, as we will see later in Section II-B.3. Note that this model is valid for hybrid datasets (in which discrete and continuous variables co-exist), since operations with discrete variables can be handled easily.

There is a close relation between MoPs and MTEs. In fact, they both arise from the same model, the Mixtures of Truncated Basis Functions (MoTBFs) [21], however the basis functions they use to represent the continuous distribution is different; MTEs uses exponentials functions, and MoPs uses polynomials. Even though they have a similar performance, it seems that MoPs have a greater accuracy when estimating probability distributions [22], however there is not a comparison between these two models for regression, in terms of prediction accuracy. This paper tries to address this issue.

2) Marginal densities: The estimation of the marginal continuos densities is carried out using the procedure in [18], which can be summarized as:

1) From the sample, obtain (x, y) where y is estimated through a kernel density estimator [23].
2) Estimate the parameters of the 1- degree polynomial $(p_1(x))$ by minimizing the MSE.
3) $p_i(x)$ may be negative in some points of the domain. Select only the most important positive part, in terms of weight or size
4) Normalize the polynomial so that its integral is equal to the proportion of points of the sample included in the selected subinterval.
5) If a part of the domain of $p_i(x)$ has been removed because of the negative values, add the necessary tails to extend the domain of the polynomial to include the whole domain of X , which yields in a piecewise function
6) Compute the corresponding MSE
7) Increase the degree
8) Repeat this procedure, and select the best polynomial.

The estimation of marginal densities is carried out for the root (goal) variable, and within the procedure for estimating conditional densities.

3) Conditional densities: The definition of the probability distributions for the features variables includes a conditional density, $f(x|c)$. If C were discrete, this conditional density would reduce to defining a different density for X for every state of the variable C. However, in the case of a continuous variable C, this conditional distribution is not so straightforward. In [24] it was shown that the only way to include C in the explicit formula of the conditional distribution for MTEs is in the definition if the domain; since MoPs belong to the same model MoTBFs than MTEs, we followed the same procedure, *i. e.*, the problem transforms to find an optimal discretization of the range of C, and for each interval in the discretization, estimate a marginal density for X.

The discretization procedure followed is an iterative procedure based on the *equal width* method:

. INPUT: Sample for X and C.
. OUTPUT: $f(x|c)$ a conditional density function.
1) Estimate $f_0(x)$ a marginal density function for X. and compute an estimation of the Mean Squared Error (MSE) of this model (see section III)..
2) Split the range of C, Ω_C in two equal width intervals.
3) In each one of the new intervals estimate $f_{11}(x)$ and f_{12} marginal density functions for X. and compute an

estimation of the MSE of this new model.

4) If the error decreases, go back to 1), try dividing every possible interval and effectively split the one with lower MSE; if not, stop and return the corresponding piecewise function.
5) Continue until the error does not decrease dividing any of the intervals, or the maximum number of intervals is reached.

This procedure is specifically designed for regression problems, since the final decision about splitting or not the domain is made according to the accuracy of the prediction in a test dataset.

III. Forecasting Results

The aim of these models is to predict as accurately as possible a goal variable.

Once the model is learnt, it is able to *predict* the goal value (c) of a given observation $\mathbf{x} = (x_1, \ldots, x_n)$ using the updated probability distribution for C :

$$f(c|\mathbf{x}) = f(c|x_1, \ldots, x_n) \propto f(c) \prod_{j=1}^{n} f(x_j|c)$$

This computation outputs a piecewise density for C. From this distribution, the point estimation of prediction of the variable is computed as the *Expected value*:

$$\hat{c} = E[C|x_1, \ldots, x_n] = \int_{-\infty}^{\infty} c f(c|x_1, \ldots, x_n) dc$$

where $x_1, \ldots, x_n$ are known values, called *evidence*.

The procedures described in the previous sections have been implemented in *R* [26], and validated through an exhaustive set of experiments. A total of 10 different datasets have been selected for the experiments from the UCI Machine Learning Repository [27] and the KEEL repository [28]. They all have both discrete and continuous features, with a wide range of number of cases (from 60 to 1030), and features (from 4 to 15). In the case of missing values, the corresponding cases were removed from the dataset.

TABLE I
DATABASES USED IN EXPERIMENTS.

Dataset	Vars	Instances
Auto	8	392
BodyFat	15	252
Cloud	8	108
Concrete	9	1030
Housing	14	506
Machine	9	209
Pollution	16	60
Servo	5	167
Strikes	7	625
Veteran	8	137

In the next sections, several hypothesis are stated, related to the performance of the MoP model in regression, and the validity of the different parameters and preprocessing methods discussed above. The error of the different methods is computed in terms of the Mean Squared Error, computed as

$$error = MSE = \frac{\sum_{i=1}^{n} (\hat{c} - c)^2}{n}$$

where n is the size of the test sample. The validation of the methods was carried out by menas of a 5-fold Cross-Validation.

A. Feature Selection method

The first issue to check is whether the feature selection scheme yields better results than including in the model every variable of the problem, and, in the case of rejection, which of the three feature selection methods obtains better results. In Table II we can see the accuracy results of the different feature selection methods, in which the goal variable range was divided at most in 3 intervals to estimate the conditional distributions.

TABLE II
COMPARISON RESULTS FOR THE DIFFERENT FEATURE SELECTION METHODS. BOLDFACED NUMBER REPRESENT THE MOST ACCURATE RESULT

Method	Auto	BodyFat	Cloud	Concrete	Housing
No	19.739	0.6566	0.6877	50964.75	35.674
Type 1	23.009	0.4012	0.5134	49618.85	27.4705
Type 2	**18.133**	**0.3897**	**0.4641**	**46683.13**	22.0975
Type 3	18.432	0.4032	0.4692	46734.49	**18.35**
Method	Machine	Pollution	Servo	Strikes	Veteran
No	6606.80	2743.18	0.8172	1578125	27613.66
Type 1	7635.33	2321.96	0.9937	324383.9	23980.95
Type 2	**3633.62**	2219.48	0.8048	**305999.5**	**23020.05**
Type 3	3823.53	**2087.88**	**0.8034**	323752.8	23662.24

This information can be graphically seen in Fig. 2, in which the *relative errors* are plotted, for a better visualization of the graph.

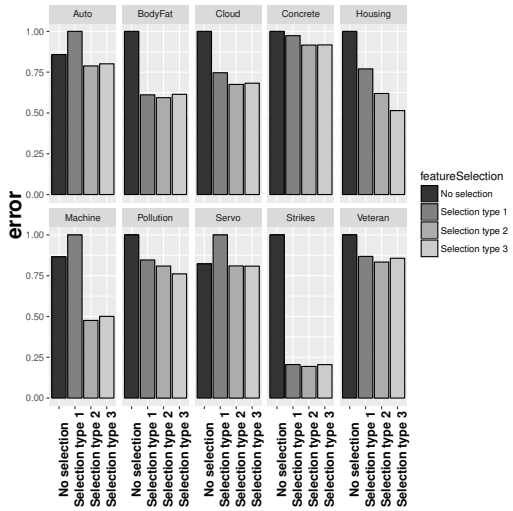


Fig. 2. Graphical representation of the prediction accuracy for the different feature selection methods

The Friedman statistical test was carried out to test if all the feature selection methods are equivalent, obtaining a p-value of $3.73e-05$, so not all the methods have similar results, in terms of accuracy. A posterior post-hoc test shows that Feature selection types 2 and 3 can be considered equivalent, but different to type 1 and no feature selection.

A box plot graph of the ranking of the different methods can be seen on Fig. 3.

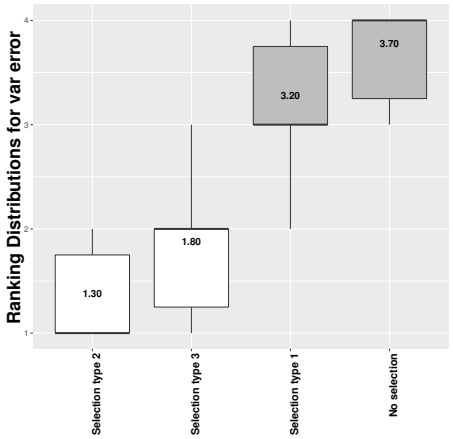


Fig. 3. Boxplot representing the ranking distribution for each feature selection method. In white color those methods obtaining better results

The conclusion we may draw from the test and the different plots is that the Feature selection method 2 is the one yielding better results.

B. Discretization method (pseudo continuous variables)

In section II-A.2 a new type of variable was defined, the pseudo continuous variable. It was shown in [18] that detecting and discretizing them was successful for discretization problems. Does it have the same impact in regression problems? To solve this issue, three new datasets were selected, since none of the ones summarized in Table I included such type of variables. The new datasets are selected from the ones in [18], which are focused on classification, however we selected a continuous variable as the goal variable, and transform the problem into a regression problem. In Table III these datasets are listed

TABLE III
DATABASES USED IN THE PSEUDO CONTINUOUS EXPERIMENTS

Dataset	Vars	Instances
Australian	15	690
Credit	16	653
German	25	1000

We can see the accuracy results of the different feature selection methods in Table IV.

This information can be graphically seen in Fig. 4, in which the *relative errors* are plotted, for a better visualization of the graph.

The Wilcoxon Signed Rank statistical test was carried out to test if detecting and discretizing pseudo continuous variables makes a difference, obtaining a p-value of 0.375, so there is not a statistical difference in terms of accuracy.

However, looking at Table IV and Fig. 4 we can argue that having pseudo continuous variables into account can benefit our results.

TABLE IV
COMPARISON OF RESULTS WHEN DETECTING PSEUDO-CONTINUOUS VARIABLES. BOLDFACED NUMBER REPRESENT THE MOST ACCURATE RESULT

Detect Pseudo continuous	Australian	Credit	German
No	104.7557	6327.3470	**420.2247**
Yes	**103.2565**	**6129.6650**	454.9360

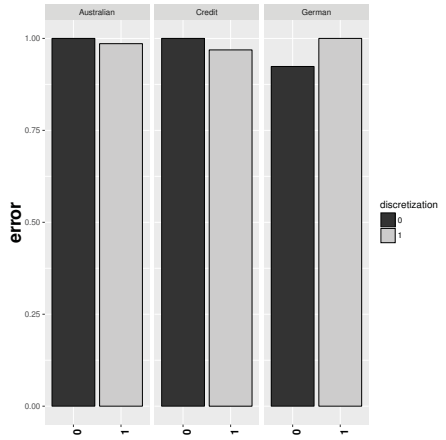


Fig. 4. Graphical representation of the prediction accuracy for when detecting and discretizing pseudo continuos variables

C. Number of intervals in conditional learning

One of the parameters that may affect most the estimation of conditional MoPs is the maximum number of intervals to split the domain of the goal variable into. In order to detect if it influences the result of the prediction, a new experiment was carried out to observe this issue, in which the Feature selection algorithm Type 2 was chosen.

In Table V we can see the accuracy results of selecting 3 or 5 intervals as maximum.

TABLE V
COMPARISON OF RESULTS FOR THE DIFFERENT INTERVALS. BOLDFACED NUMBER REPRESENT THE MOST ACCURATE RESULT

Intervals	Auto	BodyFat	Cloud	Concrete	Housing
3	18.13	0.3897	0.464	46683.13	**22.0975**
5	**17.21**	**0.368**	**0.4272**	**45930.91**	22.34
Intervals	Machine	Pollution	Servo	Strikes	Veteran
3	3633.62	2219.482	0.8048	305999.5	23020.05
5	**3486.17**	**2146.728**	**0.7148**	**292634.3**	**22811.33**

This information can be graphically seen in Fig. 5, in which the *relative errors* are plotted, for a better visualization of the graph.

The Wilcoxon Signed Rank statistical test was carried out to test if detecting and discretizing pseudo continuous

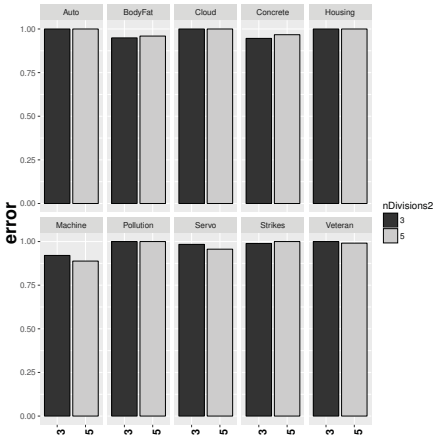


Fig. 5. Graphical representation of the prediction accuracy for different maximum number of intervals when splitting the range of the goal variable

variables makes a difference, obtaining a p-value of 0.0068, so there is a statistical difference in terms of accuracy, obtaining better results for 5 intervals than for 3 intervals. This supports the idea that, the more intervals used, the better the accuracy obtained; however the complexity of the model increases.

D. Global performance of MoP Naive Bayes model

Once the parameters and preprocessing methods affecting the MoP model estimation are checked, a global comparison with respect to some state-of-the-art methods is carried out. We have selected three methods: The MTE method within a Naive Bayes structure and a feature selection method defined in [13], as a natural competitor, the linear model, and the regression trees included in the CART model [29], both implemented in R. The MoPs model for this comparison includes feature selection type 2, and 5 intervals to split the domain of the goal variable when learning conditional distributions.

In Table VI we can see the accuracy results of selecting the different methods, including all dataset available:

TABLE VI
COMPARISON OF RESULTS FOR THE DIFFERENT METHODS. BOLDFACED NUMBER REPRESENT THE MOST ACCURATE RESULT

model	Australian	Auto	BodyFat	Cloud	
lm	118.565	**14.91**	**0.3256**	**0.22**	
MOPs	**100.01**	17.21	0.3687	0.42	
MTEs	112.36	18.82	0.411	0.84	
regTree	113.98	22.78	0.47	0.89	
model	Concrete	Credit	German	Housing	
lm	61047.23	-	1829.274	37.03	
MOPs	**45930.91**	**6078.65**	**420.224**	**18.35**	
MTEs	53610	68	502.65	40.63	
regTree	58079.17	7837.24	1827.66	35.519	
model	Machine	Pollution	Servo	Strikes	Veteran
lm	40000	2070.66	1.24	302423.5	26752.3
MOPs	**3486.17**	2087.88	0.71	292634.3	22811.3
MTEs	9820.37	**1646.33**	1.2975	**260324.4**	**15237.4**
regTree	11683.38	2300.52	**0.64**	350885.7	35181.9

This information can be graphically seen in Fig. 6, in which the *relative errors* are plotted, for a better visualization of the graph.

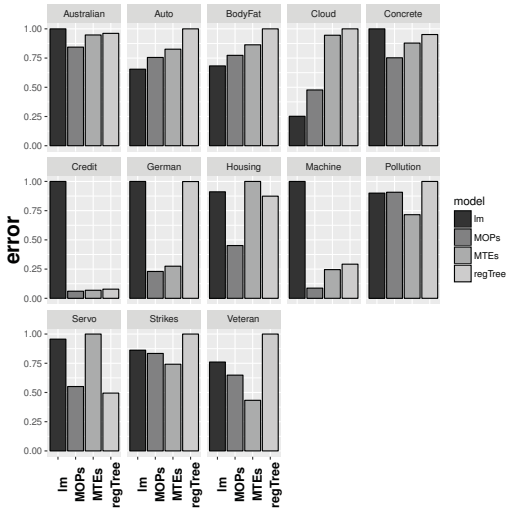


Fig. 6. Graphical representation of the prediction accuracy for the different feature selection methods

The Friedman statistical test was carried out to test if all the methods mentioned above are equivalent, in terms of accuracy, obtaining a p-value of 0.0093, so not all the methods have similar results. A posterior post-hoc test shows that all the methods have similar performance, except for regression trees, which is statistically different to MoPs model.

A box plot graph of the ranking of the different methods can be seen on Fig. 7. Even though MoPs model is not always the best model, it is clearly competitive with respect to the state-of-the-art, since there is no significative difference between their results, apart from regression trees.

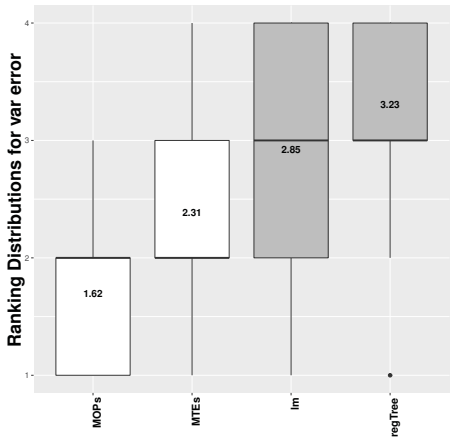


Fig. 7. Boxplot representing the ranking distribution for each feature selection method. In white color those methods obtaining better results

IV. Conclusions

We have developed a methodology for incorporating MoPs in a Bayesian network framework to solve regression models. The different parameters and preprocessing steps have been presented, as well as the learning procedures, to estimate marginal and conditional densities. Some experiments have been carried out, to look for the optimal configuration of the parameters, and the resulting method has been tested against several state-of-the-art methods, obtaining competitive results. This means that MoPs model are a valid alternative when solving a regression problem.

Some new issues can be developed in future papers, such as incorporating nee constrained structures, TAN and FAN for example, and studying the effect of this methodology in the size of the models obtained. Preliminary experiments on this issue report that MoPs models achieve similar accuracy, but incorporate a lower number of feature variables.

Acknowledgments

This work has been supported by the Spanish Ministry of Science and Innovation, through project TIN2013-46638-C3-1-P, by Junta de Andalucia through project P12-TIC-2541 and by ERDF (FEDER) funds.

References

[1] P. P. Shenoy and J. West, "Inference in hybrid Bayesian networks using mixtures of polynomials," *International Journal of Approximate Reasoning*, vol. 52, pp. 641–657, 2011.

[2] F. V. Jensen and T. D. Nielsen, *Bayesian Networks and Decision Graphs*. Springer, 2007.

[3] M. Sahami, "Learning limited dependence Bayesian classifiers," in *KDD96: Proceedings of the second international Conference on Knowledge Discovery and Data Mining*, pp. 335–338, 1996.

[4] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian network classifiers," *Machine Learning*, vol. 29, pp. 131–163, 1997.

[5] P. J. Lucas, "Restricted Bayesian network structure learning," in *Proceedings of the 1st European Workshop on Probabilistic GraphicalModels (PGM'02)* (J. Gámez and A. Salmerón, eds.), pp. 117–126, 2002.

[6] P. Domingos and M. Pazzani, "Beyond independence: Conditions for the optimality of the simple bayesian classifier," in *Proceedings of the International Conference on Machine Learning*, 1996.

[7] J. Dougherty, R. Kohavi, and M. Sahami, "Supervised and unsupervised discretization of continuous features," in *Machine Learning: Proceedings of the Twelfth International Conference* (A. P. y S. Russell, ed.), pp. 194–202, Morgan Kaufmann, San Francisco, 1995.

[8] D. Kozlov and D. Koller, "Nonuniform dynamic discretization in hybrid networks," in *Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence* (D. Geiger and P. Shenoy, eds.), pp. 302–313, Morgan & Kaufmann, 1997.

[9] U. M. Fayyad and K. B. Irani, "Multi-interval discretization of continuous-valued attributes for classification learning," in *Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI-93)*, pp. 1022 – 1027, 1993.

[10] S. Lauritzen and N. Wermuth, "Graphical models for associations between variables, some of which are qualitative and some quantitative," *The Annals of Statistics*, vol. 17, pp. 31–57, 1989.

[11] S. Moral, R. Rumí, and A. Salmerón, "Mixtures of Truncated Exponentials in Hybrid Bayesian Networks," in *Symbolic and Quantitative Approaches to Reasoning with Uncertainty* (S. Benferhat and P. Besnard, eds.), vol. 2143 of *Lecture Notes in Artificial Intelligence*, pp. 156–167, Springer, 2001.

[12] A. Fernández and A. Salmerón, "Extension of Bayesian network classifiers to regression problems," in *Advances in Artificial Intelligence - IBERAMIA 2008* (H. Geffner, R. Prada, I. M. Alexandre, and N. David, eds.), vol. 5290 of *Lecture Notes in Artificial Intelligence*, pp. 83–92, Springer, 2008.

[13] M. Morales, C. Rodríguez, and A. Salmerón, "Selective naïve Bayes for regression using mixtures of truncated exponentials," *International Journal of Uncertainty, Fuzziness and Knowledge Based Systems*, vol. 15, pp. 697–716, 2007.

[14] R. Rumí, A. Salmerón, and S. Moral, "Estimating mixtures of truncated exponentials in hybrid Bayesian network," *Test*, vol. 15, pp. 397–421, 2006.

[15] V. Romero, R. Rumí, and A. Salmerón, "Learning hybrid Bayesian networks using mixtures of truncated exponentials," *International Journal of Approximate Reasoning*, vol. 42, pp. 54–68, 2006.

[16] P. P. Shenoy, "A re-definition of mixtures of polynomials for inference in hybrid Bayesian networks," in *Symbolic and Quantitative Approaches to Reasoning with Uncertainty* (W. Liu, ed.), Lecture Notes in Artificial Intelligence 6717, pp. 98–109, Springer, 2011.

[17] M. Dash and H. Liu, "Feature selection for classification," *Intelligent Data Analysis*, vol. 1, no. 3, pp. 131 – 156, 1997.

[18] J. C. Luengo and R. Rumí, "Naive bayes classifier with mixtured of polynomials," in *Proceedings of the International Conference on Pattern Recognition Applications and Methods (ICPRAM-2015)*, pp. 14–24, 2015.

[19] P. P. Shenoy, R. Rumí, and A. Salmerón, "Some practical issues in inference in hybrid bayesian networks with deterministic conditionals," in *Proceedings of the Intelligent Systems Design and Applications (ISDA)*, 2011.

[20] R. Rumí, A. Salmerón, and P. P. Shenoy, "Tractable inference in hybrid bayesian networks with deterministic conditionals using re-approximations," in *Proceedings of the Sixth European Workshop on Probabilistic Graphical Models (PGM'2012)*, pp. 275 – 282, 2012.

[21] H. Langseth, T. D. Nielsen, R. Rumí, and A. Salmerón, "Mixtures of truncated basis functions," *International Journal of Approximate Reasoning*, vol. 53, pp. 212 – 227, 2012.

[22] H. Langseth, T. D. Nielsen, I. Pérez-Bernabé, and A. Salmerón, "Learning mixtures of truncated basis functions from data.," *International Journal of Approximate Reasoning*, 2013.

[23] J. Simonoff, *Smoothing methods in Statistics*. Springer, 1996.

[24] H. Langseth, T. D. Nielsen, R. Rumí, and A. Salmerón, "Parameter estimation and model selection for mixtures of truncated exponentials," *International Journal of Approximate Reasoning*, vol. 51, no. 5, pp. 485–498, 2010.

[25] P. L. López-Cruz, C. Bielza, and P. Larrañaga, "Learning mixtures of polynomials of multidimensional probability densities from data using b-spline interpolation," *International Journal of Approximate Reasoning*, vol. In Press, 2013.

[26] R Core Team, *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2013. ISBN 3-900051-07-0.

[27] K. Bache and M. Lichman, "UCI machine learning repository." http://archive.ics.uci.edu/ml, 2013.

[28] J. Alcalá-Fdez, A. Fernandez, J. Luengo, J. Derrac, S. García, L. Sánchez, and F. Herrera, "Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework," *Journal of Multiple-Valued Logic and Soft Computing*, vol. 17, pp. 255–287, 2011.

[29] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and regression trees*. Chapman & Hall/CRC, 1984.

Predictive Modeling for Student Retention at St. Cloud State University

Hasith Dissanayake[1], David Robinson[2], and Omar Al-Azzam[1]
[1]Department of Computer Science and Information Technology, St Cloud State University, Saint Cloud, Minnesota, 56301, USA
[2]Department of Mathematics & Statistics, St Cloud State University, Saint Cloud, Minnesota, 56301, USA
{diha1201, dhrobinson, oalazzam}@stcloudstate.edu

***Abstract*— Student graduation rates have always taken prominence in academic studies since they are considered a major factor in the performance of any university. Accurate models for predicting student retention plays a major role in university strategic planning and decision making. Students' enrollment behavior and retention rates are also relevant factors in the measurement of the effectiveness of universities. This paper provides a comparison of predictive models for predicting student retention at Saint Cloud State University (SCSU). The models are trained and tested using a set of features reflecting the readiness of students for college education, their academic capacities, financial situation, and academic results during their freshman year. Principle Component Analysis (PCA) was used for feature selection. Six predictive models have been built. A comparison of the prediction results has been conducted using all features and selected features using PCA analysis.**

Keywords*— *Retention probabilities; predictive modeling; student retention; Principal Component Analysis; Bayesian Networks; k-Nearest Neighbor; Random Forest; Artificial Neural Networks.

I. Introduction

Student enrollment behavior remains a significant focus in institutional research. Student retention is a serious national issue, and some academic areas experience it more than others. Knight et al. (2003) argued student retention became a nationwide problem in the 1980's, when retention reached 40%. This statistic suggests every 4th student leaves their college or university before graduation, and Knight called this "leakage from the engineering pipeline" [1].

While having an understanding of the reasons behind student drop-out or transfer behavior is crucial for effective enrollment management, what is even more important is the ability to predict these types of behavior to develop preventive measures. This will allow more precise tuition revenue forecasts, enrollment rate predictions, and help develop successful intervention and recruiting programs [2]. Theoretical data indicates using algorithmic approach and data mining can provide more accurate results when predicting student retention compared to traditional statistical methods [3].

This research uses data mining and an algorithmic approach to predict retention with high accuracy, while identifying the variables that could affect student dropout. Those identified variables were used to create predictive models to predict the likeliness of students' retention in their sophomore year. Considering the nature and the diversity of data available for this study, it was assumed highly accurate predictions could be made.

Data related to the admissions process was obtained from the Admissions Office at St Cloud State University (SCSU). This data was originally collected by SCSU from 2006-2010. Data related to academic records of students was obtained from the Office of Records and Registration. This dataset was extracted from ISRS in 2013 by the Office of Strategy, Planning and Effectiveness at SCSU.

The models used in this study are k-NN (k-Nearest Neighbor), Classification Tree, Random Forest, Binomial GLM, Neural Network, and Bayesian Neural Network. During the process of training the predictive models, the data was first examined for outliers and missing values. Next, these values were replaced with appropriate values to minimize the effect on incorrect predictions. Since the data was obtained through different sources, the datasets were then aggregated to one dataset linking them through the Student ID of each dataset. Out of the combined dataset, 70 variables were identified as relevant factors to student retention. At this point it was determined grouping these variables through a Principal Component Analysis (PCA) could yield better results than using the entire set of variables to train the models. However, the models were trained both ways to test the validity of this hypothesis. Finally the models were compared using their accuracy, sensitivity, specificity, positive predictive value, and negative predictive value, to identify the best model that suited the set of variables that was available.

II. Literature Review

Student retention is a widely researched area in the higher education sector, and it spans over four decades of research. Tinto (2006) states "there has also been a concomitant increase in the number of businesses and consulting firms that have sprung up, each of which claims unique capacity to help institutions increase the retention of their students." This shows the amount of research that goes into student retention in the higher education sector is large. It has eventually become a business to consult institutions on how to retain students. The 2005 National Center for Education Statistics revealed the "national rate of student retention has shown disappointingly little change over the past decade".

According to Tinto (2006), there is still much research work to be done in this field. There is a lack of translation of the research and theory into effective practice. Tinto (2006) further states before the 1970s the reason identified for low student retention rates was the failure on the part of students and not the institution. Students who dropped out were thought to be less able, less motivated, and less willing to defer the benefits of college education. Research carried out after the 1970s, mostly

by Alexander Astin (1975, 1984), Ernest Pascarella (1980), and Patrick Terenzini (1980), focused more on the environment, the institutions and the people who govern them, when determining reasons for student retention [4]. With the focus shifting to the institutions, attention to the student-faculty relationships was increased, mostly outside the classroom. Also, more focus was placed on the transition to college, by introducing extended orientation, freshman seminars, and a variety of extracurricular programs for first-year students [4]. These programs were introduced to make students feel welcome to the new culture, community or the new environment in college. Around the 1970s, institutions held the view students needed to break away from the society they were in, in order to adapt to college and thereby remain in college. But later, they discovered the gap between breaking away from society and adapting to college should be bridged through orientation programs and extracurricular programs, making them feel part of their past communities, families, churches or tribes [4].

Predictive modeling for student retention can be seen as early as 1975 with Tinto's model. Following Tinto's model, there have been many models introduced by researchers considering different factors and variables to predict student retention. Some of these models focused on identifying students with high risk of dropping out from college [5] [6]. Furthermore, Alkhasawneh & Hargraves (2014) credit Ben Gaskin (2009), with traditional methods of statistical analysis such as logistic regression being used to predict student retention. In most recent years, data mining, which is recognizing patterns in large data sets and then understanding those patterns, has been used to study student retention because of high accuracy and the robustness of missing data [5].

Tinto's model considered social and academic impacts on a student's decisions (voluntarily or involuntarily) to drop out from college. The model is based on Durkheim's (1961) theory of suicide, especially its notions of the cost-benefit analysis of individual decisions about investment in alternative educational activities, which comes from the field of economics of education. Tinto (1975) makes the connection with Durkheim's suicide theory by considering the case of dropping out as committing suicide and views college as a social system. He then relates all the reasons behind committing suicide to that of dropping out of a college when it is viewed as a social system. Furthermore, as colleges also consist of an academic system, he combines the academic factors to the model to be more effective and to shape it to be more suitable to the college structure [7]. These factors are valid even for today's college structure and should be considered as inputs in the predictive models even today.

In addition to Tinto's ideas in 1975, Astin (1993), in his I-E-O (Input-Environment-Output) model, suggests in addition to factors affecting student retention during their college life, researchers should also explore factors affecting student retention before entering college, such as race/ethnicity, gender, family background (which he calls "pre-college characteristics"), high school GPA, and student self-reported data. Astin discusses how these factors affect one's academic and social life while that person is in college and how it could affect the decision she/he makes [8]. Other than the factors Tinto focused on in his research, the factors Astin discusses in his research also could be variables that might change the outcome of a retention model and make it more accurate.

This study is an attempt to identify factors contributing to student retention, primarily taking into consideration the models of both Tinto (1975) and Astin (1993), data obtained from the SCSU Admissions Office on student admission and first year grades, and data from first year student surveys. Knowledge and research of time to degree (TTD) completion remains one of the blind spots of student retention studies. A number of models have been developed to build successful predictions: k-NN, decision trees, Bayesian networks and neural networks, among others [9].

III. Research Methodology

When conducting a study to predict student retention in colleges it is important to choose the appropriate input data and variables, as well as a modelling framework. The main variables used to predict student retention can be categorized into the following groups: financial, encouragement from friends and family, college/university integration, ethnicity [1], academic performance, social integration, institutional commitment, goal commitment, and persistence [10]. Accounting for all of the important variables is crucial for building a successful model with a strong prediction power. Herzog (2006) identified the institutional support factor as a key element in predicting student retention, while Skhakil (2002) observed differences in behavior among commuters and resident students and proposed a degree of connectedness as a most likely explanatory variable.

A. Predictive models and their accuracy

Comparison of the efficiency and accuracy of various methods to predict retention rate and TTD has been performed by multiple studies, revealing contradictory results. Luan (2002) suggests the decision tree analysis method is a better predictor of community college student transfer rate, when compared to neural network analysis. It is important to identify a specific research goal and have a good understanding of the available data type, prior to choosing a specific method [9]. Neural network models have been criticized for poor performance, however, they are widely utilized for data mining (see Table 1) and are considered to have a strong predictive power [14].

Kabakchieva (2012) has simultaneously compared three models (neural network, decision trees and k-NN) to evaluate their prediction power when studying student performance. The highest accuracy was observed when using the neural network model (73.59%), followed by the decision tree demonstrating 72.74% accuracy. The k-NN model resulted in 70.49% accuracy [12].

TABLE I. COMPARISON OF ARTIFICIAL NEURAL NETWORKS (ANN), TREES, K-NN AND BAYESIAN NETWORKS MODELS (AFTER: HASTIE, TIBSHIRANI & FRIEDMAN, 2009)

Characteristics	ANN	Trees	k-NN	BNN
Handling of mixed type data	poor	good	poor	good
Handling missing values	poor	good	good	good
Robustness of outliers	poor	good	good	good
Computational scalability	poor	good	poor	good
Handling irrelevant input data	poor	good	poor	medium
Interpretability	poor	medium	poor	medium
Predictive power	good	poor	good	good
Extracting linear combination of features	good	poor	medium	medium

Decision trees are considered one of the best "off-the shelf" methods of data analysis in terms of speed, interpretability, computational scalability, etc. [14].

B. Data used in this research

The data used in this research was obtained through the Admissions Office and the Office of Strategy, Planning, and Effectiveness at SCSU. This data includes the following categorical data which was used to identify the input variables for the models. The following subsections use example variables from the set of variables available, to better explain the categorization.

1) Readiness for a college education: The first category we considered was the likelihood of student retention in school relative to the applicant's level of readiness when applying for college. For example, the variable *AdmitDaysBeforeTerm* can be used as a parameter to measure this likelihood. It would entail the number of days starting from the day the student receives admission to college, to the academic year start date. It is possible to assume the earlier the student gets admission, the earlier s/he would have started the application process. And this would help us conclude that the student's decision to get a college education was well thought out.

2) Academic capacity: The variable *ACE_Student*, identifies a student who is provisionally admitted and needs more preparation for college studies. This can be used to determine the academic capacity of students. It is more likely that students who are not provisionally admitted will perform better in their college education. This would lead to academic satisfaction; a factor directly linked to student retention at any stage.

3) Financial stability: There are a few variables indicating financial stability of the student. Apart from the variables related to Federal student aid or scholarships provided by the college, there are other variables such as, *MilesToSCSU,* that is, the number of miles to SCSU from the applicant's home. If a student thinks that s/he is too far from school, in which case the student must rent an apartment closer to the school or seek university housing, this could impose additional financial stress on the student, and could eventually lead to the student dropping out from college.

4) Other variables: These are variables that do not represent any of the above mentioned categories, but are still important for the models. Some examples are as follows.

a) ClosestToSCSU: This parameter determines if the closest college to the applicant's home is SCSU or not. This could later have an effect if the student decides to transfer to a closer school. In that case, SCSU fails to retain the student since the student has another school in closer proximity than SCSU.

b) FirstGenStudent: Being a first generation college student could have a negative effect on continuing college studies. Reasons for this could be lack of guidance or motivation from family.

c) FAFSA number: This is the number applicants mark SCSU in the precedence order of the FAFSA list. If this number is low, it is likely that the student will try to transfer to a school which s/he prefers.

d) Age: The student's age could be a determining factor when deciding if s/he wants to continue studying. It is highly likely the student would have commitments and responsibilities other than her/his education, the older in age the student is. This could be another reason why adult students drop out from college.

C. Data cleaning process

A data cleaning process was conducted using the following steps:

- Performed data type conversion (character to numeric, character to factor)
- Aggregated data to student ID level
- Identified outliers and removed them appropriately
- Cleaned/replaced missing values
- Merged data into one dataset

1) Identifying outliers: The following variables were identified as ones with outliers and were treated accordingly.

a) ACT_Composite scores: Outliers were replaced by the mean value of the said dataset taken without the outliers.

b) MilesToSCSU: Outliers were rectified using a log function on the variable. Before the log function was calculated, all the data points were increased by one to avoid log(0) resulting in NA values after the calculation.

c) HousingAppDaysBeforeTerm: This variable had outliers both from positive and negative sides. Outliers were replaced with the maximum values of either sides.

d) AdmitDaysBeforeTerm: All the outliers were replaced with the maximum value of the dataset taken without the outliers.

e) T1_TermCumulativeLocalCreditsEarned: The outliers which were identified for this variable were replaced by the maximum value of the dataset when taken without the outliers.

f) T2_TermCumulativeLocalCreditsEarned: The outliers which were identified for this variable were replaced by the maximum value of the dataset when taken without the outliers.

2) Cleaning missing values: The variables which had missing values and the respective number of missing values are shown in the Table 2.

TABLE II. NUMBER OF MISSING VALUES FOR EACH VARIABLE IN THE DATASET

Variable	Number of missing values
ACT_Composite	5456
HS_GPA_4Scale	4179
HSPct	5653
QPP	5
EnglTotal	3229
ClosestToSCSU	464
MilesToSCSU	464
HousingAppDaysBeforeTerm	9044
FAFSADaysBeforeTerm	2690
FAFSADayOfYear	2690
FAFSA_Nbr	2690
TotalCRHR_TRSF	8312
TransferGPA	8484
TransferQP	8312
HS_MnSCURegion	502
T1_TermGPA	299
T2_TermGPA	2502
T2_TermLocalCreditsAttempted	2249
T2_TermLocalCreditsEarned	2249

a) QPP: The missing values of this variable was replaced with the mean value of the QPP dataset.

b) ClosestToSCSU: Since this is a Boolean variable all the missing values were considered as not reported and substituted with a zero (false).

c) MilesToSCSU: After considering the retained records out of these missing value records, the maximum value of the dataset was substituted for the missing values.

d) FAFSADaysBeforeTerm: The effects of the records with missing values in the *FAFSADaysBeforeTerm* variable on retention shows 69.5% of the time a student was retained when a value was present for *FAFSADaysBeforeTerm* variable. The same number was at 75.7% when the variable had a missing value.

Furthermore, by looking at the boxplot of the *FAFSADaysBeforeTerm* vs *T3_Enrolled* which is shown in Fig. 1, it can be concluded the *FAFSADaysBeforeTerm* variable has very low impact on retention.

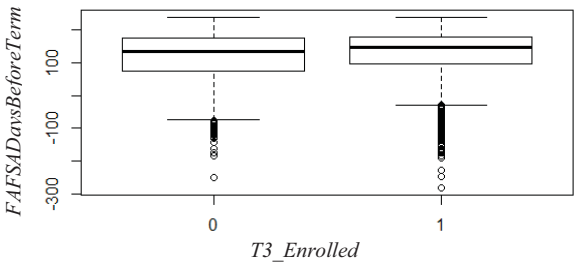


Fig. 1. Boxplot of the *FAFSADaysBeforeTerm* vs *T3_Enrolled*

Considering these facts, it can be assumed that using a fixed value for missing values, rather than a statistic calculated from the sample (like mean), can make the results more robust. This is because when a new dataset is processed, it will have the same replacement values for the missing data, and thus will make the prediction more consistent.

All the variables that had missing values related to FAFSA behaved in an identical way. Therefore, *FAFSADayOfYear* and *FAFSA_Nbr* were also treated the same way as *FAFSADaysBeforeTerm.* Missing values were replaced with a zero.

e) EnglTotal: Whenever there is no value for *High school English subject total*, -1 is assigned to that data point. Hence all the missing values were treated as instances, the student did not have an English score to report and were given a value of -1.

f) HS_GPA_4Scale, ACT_Composite, HSPct(High School Percentage): These missing values were replaced by mean value of the available dataset of the same variable.

g) Other variables: The remaining variables could be considered as instances where data were not reported. These variables were replaced with a value of zero (0).

D. Building Models

Once the dataset was cleaned, there were 68 variables excluding Student ID and T3_Enrolled variables, meaning there were 68 covariates that could be used as predictors with T3_Enrolled being the target variable. The distribution of the target variable is shown in Table 3.

TABLE III. DISTRIBUTION OF T3_ENROLLED VARIABLE

T3_Enrolled	Frequency	Rate
TRUE	10968	0.7067
FALSE	4551	0.2933

1) Preprocessing: Seventy five percent (75%) of the available data was used to train the selected model. The other 25% of the data was used as the test dataset to verify the validity of the final trained model. Choosing a model was done by comparing the number of models for their accuracy, sensitivity, specificity, positive predictive value, and negative predictive value. In the comparison stage, the models were built and tested using the same sample dataset.

2) Identifying near-zero variance variables: In some situations, the data has predictors that only have a single unique value (i.e. a "zero-variance predictor"). For many models (excluding tree-based models), this may cause the model to crash or the fit to be unstable. Similarly, predictors might have only a handful of unique values that occur with very low frequencies. These predictors may become zero-variance predictors when the data are split into cross-validation/bootstrap sub-samples or when a few samples may have an undue influence on the model. These "near-zero-variance" predictors may need to be identified and eliminated prior to modeling[24]. After removing the near zero variance variables, there were 64 variables left as predictors.

3) Principal Component Analysis: In some cases, there is a need to use principal component analysis (PCA) to transform the data to a smaller sub–space where the new variables are uncorrelated with one another. Using a number of correlated predictors may lead to over fitted models. PCA transformation eliminates this problem as it results in uncorrelated variables which will improve the results of the predictions.

Principal Component Analysis is carried out usually to achieve the following:

- Extract the most important information from the data table.
- Compress or reduce the size of the data set by only keeping this important information.
- Simplify the description of the data set.
- Analyze the structure of the observations and the variables.

When Principal Component Analysis is carried out, a new set of variables is computed. These new variables, called Principal Components, are attained as linear combinations of the original variables[25].

After removing the near-zero variance variables from the data set, a PCA was performed to compress the data set with a threshold of 95%. After the Principal Component Analysis, the number of variables were reduced from 64 to 35.

4) Training Models: Once the preprocessing of data was done, each of the models were built to use in the comparison stage. The models which were built in this phase were KNN (K Nearest Neighbor), Classification Tree, Random Forest, Binomial GLM, Neural Network, and Bayesian Neural Network. After building each model, accuracy, sensitivity, specificity, positive prediction value, and negative prediction value was calculated using the confusion matrix (refer comparing models phase for description of these terms).

Building models were done in two different phases. First, the models were built using PCA compressed data, and secondly using the entire variable set. These two phases were done to compare and identify which model, using which set of variables, had the best statistics. It was also used to test the hypothesis from before, the Principal Component Analysis should improve the results of prediction.

IV. Experimental Results

A. Comparing Models

The results obtained during the model building phase were used in this stage to determine if the hypotheses made about using PCA will result in better predictive models. Secondly, the best model was selected by looking at the statistics of the trained models.

1) Measures used to compare models: A few measures were used to compare the models: accuracy, sensitivity, specificity, positive predictive value, and negative predictive value.

a) Confusion Matrix: Confusion matrix is used to measure the performance of a predictive model. These are often used in classification models. A confusion matrix is composed of true positives, true negatives, false positives, and false negatives, which are statistics calculated using predicted values and actual values.

TABLE IV. Example of a confusion matrix

n=1000		**Prediction**	
		FALSE	TRUE
Actual	FALSE	175	35
	TRUE	115	675

b) Accuracy: Accuracy is how probable it is on average for the prediction of the model to be correct. This is calculated as the proportion of the number of correctly classified cases to the total number of cases. Correctly classified cases are the total of true positives and true negatives. Equation (1) represents the calculation of accuracy, n being the total number of cases.

$$Accuracy = \frac{True\ Positives + True\ Negatives}{n} \quad (1)$$

c) Sensitivity: Sensitivity is how probable it is to classify a case as true when it is actually true. This is calculated as the proportion of correctly classified true cases within actual true cases. Equation (2) represents the calculation of sensitivity.

$$Sensitivity = \frac{True\ Positives}{Actual\ Trues} \quad (2)$$

d) Specificity: Specificity is how probable it is to classify a case as false when it is actually false. This is calculated as the proportion of correctly classified false cases within actual false cases. Equation (3) represents the calculation of specificity.

$$Specificity = \frac{True\ Negatives}{Actual\ Falses} \quad (3)$$

e) Positive Predictive value: Positive predictive value is how probable it is for a case to be actually true when the model predicts it to be true. In the context of this thesis problem, the probability would be how likely it is that a student will retain, when the model identifies the student as a student who will retain. This is calculated as the proportion of correctly classified true cases within predicted true cases. Equation (4) represents the calculation of positive predictive value.

$$Pos.\ Pred.\ Value = \frac{True\ Positives}{Predicted\ Trues} \quad (4)$$

f) Negative Predictive Value: Negative predictive value is how probable it is for a case to be actually false when the model predicted it to be false. In the context of this thesis problem, it is how likely it would be for a student to dropout, when the model identifies the student as a student who will dropout. This is calculated as the proportion of correctly classified false cases within predicted false cases. Equation (5) represents the calculation of negative predictive value.

$$Neg.\ Pred.\ Value = \frac{True\ Negatives}{Predicted\ Falses} \quad (5)$$

2) Statistics of trained models: Trained models were tested using the same sample dataset. Table 5 and 6 display the statistics of the results.

TABLE V. CALCULATED STATISTICS OF PREDICTIONS OF MODELS USING PRINCIPAL COMPONENT ANALYSIS STRUCTURE

	Accuracy	Sensitivity	Specificity	Pos.pred. val	Neg.pred. val
KNN	0.8337	0.9634	0.5139	0.8301	0.8506
Classification Tree	0.8136	0.8915	0.6215	0.8531	0.6992
Random Forest	0.8587	0.9535	0.625	0.8624	0.8451
Binomial GLM	0.8307	0.9352	0.5729	0.8437	0.782
Neural Network	0.8487	0.9493	0.6007	0.8542	0.8277
BNN	0.8527	0.9577	0.59375	0.8532	0.8507

TABLE VI. CALCULATED STATISTICS OF PREDICTIONS OF MODELS USING ORIGINAL SET OF VARIABLES

	Accuracy	Sensitivity	Specificity	Pos.pred. val	Neg.pred. val
KNN	0.7405	0.9437	0.2396	0.7536	0.633
Classification Tree	0.8357	0.9563	0.5382	0.9362	0.8334
Random Forest	0.8477	0.9479	0.6007	0.8541	0.8238
Binomial GLM	0.8307	0.9324	0.5799	0.8485	0.7767
Neural Network	0.8407	0.9338	0.6111	0.8555	0.7892
BNN	0.8507	0.9507	0.6042	0.8555	0.8325

The KNN algorithm had an accuracy improvement of 0.0932 when the PCA structure was used to train the model. Particularly, the specificity of KNN when using PCA was improved substantially. Improvement of specificity was 0.2743. That is a 27% increase when PCA was used. Sensitivity was also improved by 0.0197.

The Classification Tree did not show an overall performance improvement when the PCA structure was used. However, specificity was 0.0833 higher when PCA was used whereas sensitivity was 0.0648 higher when the original set of variables was used. Overall, there was a 0.0221 or 2.21% improvement of accuracy when the original set of variables was used to train this model.

Random Forest indicated higher performance in all measures when the PCA structure was used for training the model. It had a 0.0056 higher sensitivity value and a 0.0243 improvement of the specificity value. The accuracy improvement was at 0.011 when the PCA structure was used. Even though it did not show drastic improvements when the PCA structure was used, considering all measures had higher values than using the original set of variables, it is logical to determine Random Forest had higher performance when the PCA structure was used.

Binomial GLM did not have any difference in accuracy value in either cases. Even though sensitivity was 0.0028 higher when PCA was used, specificity was reduced by 0.007. Overall, this model did not show any significant difference when the PCA structure was used over the original set of variables.

Neural Networks showed improvement in both accuracy and sensitivity with 0.012 and 0.007 respectively when PCA structure was used. Yet, specificity dropped by 0.0104. Overall, the Neural Network model performed well when the PCA structure was used.

BNN model also behaved in a similar way to Neural Network model. It had higher accuracy and sensitivity with improvements of 0.002 and 0.007 respectively, but, specificity decreased by 0.0105. The overall performance was better when the PCA structure was used to train BNN.

V. CONCLUSION

By comparing the statistics in Table 5 with the same statistics in Table 6, it can be concluded there are no clear cut results to confirm the hypothesis made about the principal

component analysis always being true. Or that PCA yields better results than the results obtained by using the original set of variables. However, when the accuracy value is compared, it is evident in most cases, models built using the PCA structure yielded better results. But, in some cases, the models built using the original set of variables were better. Namely, KNN had the most significant difference using the PCA structure, while Random Forest had a slight advantage over the original variable list. Binomial GLM, Neural Network, and BNN had very little to no advantage using the PCA structure. The Classification Tree model had a considerable advantage when the original set of variables was used over the models trained using PCA structure.

Apart from the Classification Tree model, all the other cases had improvements of sensitivity when the PCA structure was used in training models. Half the models showed improvements in specificity when the PCA structure was used. Namely, KNN, Classification Tree, and Random Forest had better specificity values with PCA while Binomial GLM, Neural Networks and BNN had dropped in the same category.

Out of the six models that were used, Random Forest was the only model that showed improvement in all areas when the PCA structure was used. It was also the model with the highest accuracy value. The overall average of accuracy of all models was approximately 84%. Binomial GLM showed the lowest accuracy at 83.07%, while Random Forest showed the highest accuracy at 85.87% when the PCA structure was used.

An analysis of the results reveal Random Forest and BNN had the highest accuracy values and they were very close to each other. Both models had the highest accuracy when the PCA structure was used. However, BNN had a slightly higher positive predictive value when the model was trained using the original set of variables. It can be concluded either of these models can be used with PCA data structure to predict retention at St Cloud State University with a very high accuracy. Furthermore, it can be concluded the use of data mining and an algorithmic approach to predict retention can yield results with high accuracy.

REFERENCES

[1] Knight, D. W., Carlson, L. E., & Sullivan, J. F., Staying in Engineering: Impact of a Hands-On, Team-Based, First-Year Projects Course on Student Retention. American Society for Engineering Education Annual Conference & Exposition. Session 3553. 2003.

[2] Richardson, J. & Dantzler, J., Effect of a Freshman Engineering Program on Retention and Academic Performance. Frontiers in Education Conference Proceedings, ASEE/IEEE, pp. S2C-16-S2C-22. 2002.

[3] Education – US News & World Report. "Freshman Retention Rate". Retrieved from http://colleges.usnews.rankingsandreviews.com/best-colleges/rankings/national-universities/freshmen-least-most-likely-return

[4] Tinto, V., "Research and practice of student retention: What next?". Journal of College Student Retention: Research, Theory and Practice, 8(1), 1-19. 2006.

[5] Alkhasawneh, R., & Hargraves, R. H., "Developing a Hybrid Model to Predict Student First Year Retention in STEM Disciplines Using Machine Learning Techniques". Journal of STEM Education: Innovations & Research, 15(3), 35-42. 2014.

[6] Herzog, S., Estimating Student Retention and Degree-Completion Time: Decision Trees and Neural Networks Vis-à-Vis Regression. New directions for institutional research, 131, pp. 17-33. 2006.

[7] Tinto, V., "Dropout from Higher Education: A Theoretical Synthesis of Recent Research". Review of Educational Research, (1). 89. 1975.

[8] Astin, A. W. (1993). Assessment for excellence: the philosophy and practice of assessment and evaluation in higher education. Phoenix, AZ: Oryx Press.

[9] Bogard, M., Helbig, T., Huff, G., & James, C. A comparison of empirical models for predicting student retention. White paper. Office of Institutional Research, Western Kentucky University. 2011.

[10] Cabrera, A., Nora, A., Castaneda, N., College persistence: Structural equations modeling test of an integrated model of student retention. The Journal of Higher Education, 64(2), 123–139. 1993.

[11] Yadav, S. K., Bharadwaj, B., & Pal, S., Mining Education data to predict student's retention: a comparative study. arXiv preprint arXiv:1203.2987. 2012.

[12] Kabakchieva, D., Student performance prediction by using data mining classification algorithms. International Journal of Computer Science and Management Research, 1(4), 686-690. 2012.

[13] Cover, T. M., & Hart, P. E., Nearest neighbor pattern classification.Information Theory, IEEE Transactions on, 13(1), 21-27. 1967.

[14] Hasti, Tibshirani and Friedman. Elements of Statistical Learning: Data Mining, Inference, and Prediction. Second Edition. Springer-Verlag. 2009.

[15] Arifovic, J., & Gencay, R., Using genetic algorithms to select architecture of a feedforward artificial neural network. Physica A: Statistical mechanics and its applications, 289(3), 574-594. 2001.

[16] Alkhasawneh, R., & Hobson, R., Modeling student retention in science and engineering disciplines using neural networks. In Global Engineering Education Conference (EDUCON), 2011 IEEE (pp. 660-663). IEEE. 2011.

[17] Yang, J., Ma, J., Berryman, M., & Perez, P., A structure optimization algorithm of neural networks for large-scale data sets. In Fuzzy Systems (FUZZ-IEEE), 2014 IEEE International Conference on (pp. 956-961). IEEE. 2014.

[18] Brown, J. L., "Developing a freshman orientation survey to improve student retention within a college". College Student Journal, 46(4), 834-851. 2012.

[19] CollegeBoard Advocacy. "How Colleges Organize Themselves to Increase Student Persistence: Four-Year Institutions". Retrieved from https://professionals.collegeboard.com/profdownload/college-retention.pdf, April 2009.

[20] Student Financial Aid Services Inc. "What is FAFSA?" Retrieved from http://www.fafsa.com/understanding-fafsa/what-is-fafsa

[21] Luan, J., & Serban, A., Knowledge Management: Building a Competitive Advantage in Higher Education. New Directions for Institutional Research, 113. San Francisco: JosseyBass, 2002.

[22] Skahill, M., The role of social support network in college persistence among freshmen students. Journal of College Student Retention, 4(1), 39-52, 2002.

[23] U.S. Department of Education. "Federal Pell Grant Program". Retrieved from http://www2.ed.gov/programs/fpg/index.html

[24] Kim, J.-H.. Estimating classification error rate: Repeated cross-validation, repeated hold-out and bootstrap. Computational Statistics and Data Analysis, 53(11), 3735–3745. doi:10.1016/j.csda.2009.04.009

[25] H. Abdi and L.J. Williams, "Principal Component Analysis", Wiley Interdisciplinary Reviews: Computational Statistics, 2010.

The Generalized Shortest Path Kernel for Classifying Cluster Graphs

Linus Hermansson
Department of Mathematical and Computing Sciences,
Tokyo Institute of Technology, Meguro-ku Ookayama, Tokyo 152-8552, Japan

***Abstract*—We consider using an SVM together with graph kernels in order to classify graphs into classes. Although several graph kernels exist and have been shown experimentally to work for certain graph classification problems, it is often difficult to theoretically analyze for which graph classification problems a particular graph kernel will work well. We provide a semi-theoretical analysis of the feature vectors of the newly published generalized shortest path kernel, which results in a conjecture about the accuracy of an SVM which uses the generalized shortest path kernel. We back up our conjecture with experimental results, for a classification task where the goal is to classify if a graph contains k or $k+1$ number of clusters. Keywords: SVM, Graph Kernel, Machine Learning.**

1 Introduction

A graph kernel can be seen as a similarity measure between graphs, this similarity measure can be used together with a *support vector machine* (SVM) in order to be able to classify graphs into different classes [3], [8], [9]. Graph classification has many useful applications and by solving this problem it is possible to for example classify if human tissue contains cancer or not [1].

One particular type of graphs which comes up in many applications are cluster graphs. By a cluster in a graph we mean a subgraph that contains a higher density of edges inside the clusters than between clusters. Cluster graphs can represent for instance social networks, where each node represents a person and each edge represents a friendship relation. For such graphs the clusters correspond to groups of friends.

The *shortest path* (SP) kernel is a very popular graph kernel to use in combination with an SVM for graph classification and has been shown to be able to classify a wide variety of graphs, such as for instance classifying protein graphs by their enzyme class [3] or classifying if a graph represents a low-density-parity-check code [9]. The newly introduced *generalized shortest path* (GSP) kernel [7], when used by an SVM classifier, has been shown to outperform the SP kernel at classifying certain types of graphs, even though calculating the feature vectors for the GSP kernel takes almost the same amount of time as the SP kernel when using standard algorithms. In this paper we discuss in more detail why and when the SVM classifier, which uses the GSP kernel, outperforms the SVM classifier which uses the SP kernel. For this purpose, we consider harder cluster graph classification problems than the one discussed in [7], that is, for each k, the problem of distinguishing whether a given graph consist of k-clusters or $k+1$-clusters. The hardness of this problem is determined by four parameters: p, the probability of having an edge between two nodes that are inside the same cluster; q_1 (or q_2), the probability of having an edge between two nodes that are in different clusters, and k, the number of clusters. We investigate how the accuracy of the SVM classifier which uses the GSP kernel relates to these four parameters. The details of this classification problem can be found in Sect. 3.

This work is supported in part by KAKENHI No. 24106008.

Our experimental results show that for certain parameter values, the SP kernel does not work at all, even though the GSP kernel completely solves the problem. This happens when p is large and k is small. The reason that the SP kernel does not work well is most likely due to the fact that for these parameters, all node pairs are at very close distances, making it difficult to distinguish feature vectors from the different classes of graphs for the SP kernel.

For understanding the GSP based SVM classifier, we think that it is important to give some formula estimating its accuracy. Unfortunately, though, there are several parameters even for our relatively simple classification task, and the mechanism of the GSP based SVM classifier is not so simple, it seems difficult to give a rigorous analysis of its performance. Thus, we took a heuristic approach in this paper. Based on observations from our experimental results, we propose some statistical value as a "hidden parameter" that determines the accuracy of our classifier. We then derive, as our main technical contribution, some evidence supporting this conjecture using our experimental results.

The rest of the paper is organized as follows. Section 2 contains definitions and terms used throughout the paper. In Sect. 3 we define the random models used to generate graphs and formally define our graph classification problem. Section 4 defines the graph kernels which we investigate. Section 5 contains our semi-theoretical analysis. Section 6 contains our experimental results and Sect. 7 contains our conclusions.

2 Preliminaries

In this section we define our notations that will be used throughout the paper. For a fixed graph, G,V and E denote the graph, the set of vertices and set of edges respectively. We denote the number of nodes and edges in a particular graph by the symbols n and m.

Since our approach is based on graph kernels, (see Sect. 4 for details) which counts the number of node pairs in

graphs that are at particular distances and have a particular number of shortest paths, we define necessary notations for the feature vectors of these graph kernels so that we can appropriately discuss the relevant graph properties. We denote the number of node pairs, that have a shortest path of length $d \geq 1$, in a graph G, by n_d. For $d, x \geq 1$, by $n_{d,x}$, we denote the number of node pairs that are at distance d and have x number of shortest paths. For any given graph G, We call a vector $\boldsymbol{v}_{\rm sp} = [n_1, n_2, \ldots]$ a *SPI feature vector*. For any given graph G, We call a vector $\boldsymbol{v}_{\rm gsp} = [n_{1,1}, n_{1,2}, \ldots, n_{2,1} \ldots]$ a *GSPI feature vector*. Note that $n_d = \sum_x n_{d,x}$. We sometimes in our analysis use feature vectors where we consider shortest paths from a fixed node in a graph, instead of all node pairs. Whenever we use such a version of the feature vectors we point it out in the text.

In this paper we consider different random models for generating graphs. For any specific such model, $\mathrm{E}[\,\boldsymbol{v}_{\rm sp}\,] = [\mathrm{E}[n_1], \mathrm{E}[n_2], \ldots]$, denotes the expected SPI feature vector. We denote the expected GSPI feature vector by $\mathrm{E}[\,\boldsymbol{v}_{\rm gsp}\,] = [\mathrm{E}[n_{1,1}], \mathrm{E}[n_{1,2}] \ldots, \mathrm{E}[n_{2,1}] \ldots]$. In order to separate k and $k+1$-cluster graphs, we use a super script $^{(k)}$ or $^{(k+1)}$. So that for instance $n_{2,1}^{(k)}$ is the number of node pairs that have a shortest path of length 2 and exactly 1 shortest path, *in a k-cluster graph*. We also consider nodes from particular clusters, we define that $n_d^{(y,k)}$ is the number of nodes at distance d, in a k cluster graph, *that are in cluster y only*.

We compare the performances for an SVM that uses either the SP kernel or the GSP kernel. Sometimes for simplicity we write "the accuracy of the SP/GSP kernel", when we mean the accuracy of an SVM which uses the SP/GSP kernel. When showing figures of average experimental feature vectors, the average is always over 200 i.i.d. feature vectors.

3 Random graph model and our graph classification task

The problem that we analyze in this paper is the problem of classifying random graphs as having either k or $k+1$ number of clusters, where k is an integer greater than or equal to 2. A cluster is a subgraph of a graph where the edge density is higher than between the clusters. The graphs which we use in our experiments are generated according to the *planted partition model* [10], which is an extension of the well known *Erdős-Rényi* model [2]. In the planted partition model, for a k-cluster graph, all nodes belong to one of k clusters. Each cluster contains n/k number of nodes. The presence of an edge between any node pair is determined randomly. Two nodes that are from the same cluster are connected with the probability p, while nodes that are from different clusters are connected with probability q_1. Where we define q_1 as

$$q_1 = p(1-\beta).$$

with parameter β which is one of the key parameters in our experiments. We denote the model for generating such graphs as $G(n, k, p, q_1)$.

Each dataset consists of k-cluster graphs and $k+1$-cluster graphs, For the $k+1$-cluster graphs, two nodes that are from the same cluster are connected with probability p, which is the same probability as for the k-cluster graphs. Two nodes that are from different clusters are connected with probability q_2. In order that it is not possible to simply distinguish the different graph types based on the number of edges, we fix q_2 so that the expected number of edges in the k-cluster graphs and the $k+1$-cluster graphs are the same. It is easy to verify that this implies that we have to choose q_2 as

$$q_2 = q_1 + \frac{1}{k^2}(p - q_1) = q_1 + p\frac{\beta}{k^2}.$$

We call the model for generating such graphs $G(n, k+1, p, q_2)$.

The problem which we consider in this paper is to train an SVM using graphs from both models (for a fixed set of parameters p, q_1, q_2, k). We then want to be able to use this SVM to predict, for random graphs which are generated from one of the two models, which of the two models was used to generate each particular graph. I.e. we want our SVM to correctly be able to classify, after training, if a given random graph is a k-cluster graph or a $k+1$-cluster graph.

4 Shortest path kernel based SVM and generalized shortest path kernel based SVM

Here we define the relevant graph kernels which we use together with an SVM in order to classify graphs from our classification problem defined in Sect. 3. A graph kernel is a function $k(G_1, G_2)$ on pairs of graphs, which can be represented as an inner product $k(G_1, G_2) = \langle \phi(G_1), \phi(G_2) \rangle_{\mathcal{H}}$ for some mapping $\phi(G)$ to a Hilbert space $\mathcal{H}$, of possibly infinite dimension. It is convenient to think of graph kernels as similarity functions on graphs. Graph kernels have been used as tools for SVM classifiers for several graph classification problems [3], [4], [8].

The *shortest path* (SP) kernel, compares graphs based on the shortest path length of all pairs of nodes [3]. Let $D(G)$ denote the multi set of shortest distances between all node pairs in the graph G. For two given graphs G_1 and G_2, the SP kernel is then defined as:

$$K_{\rm SP}(G_1, G_2) = \sum_{d_1 \in D(G_1)} \sum_{d_2 \in D(G_2)} k(d_1, d_2),$$

where k is a positive definite kernel [3]. One of the most common kernels for k is the indicator function, as used in [3]. This kernel compares shortest distances for equality. Using this choice of k we obtain the following definition of the SP kernel:

$$K_{\rm SPI}(G_1, G_2) = \sum_{d_1 \in D(G_1)} \sum_{d_2 \in D(G_2)} \mathbb{1}\left[d_1 = d_2\right].$$

We call this the *shortest path index* (SPI) kernel. It is easy to check that $K_{\rm SPI}(G_1, G_2)$ is simply the inner product of the SPI feature vectors of G_1 and G_2.

The *generalized shortest path* (GSP) kernel, is defined using the shortest path length and number of shortest paths between all node pairs. For a given graph G, by $ND(G)$ we denote the multi set of numbers of shortest paths between all node pairs of G. The GSP kernel is

then defined as:

$$K_{\mathrm{GSP}}(G_1, G_2) = \sum_{d_1 \in D(G_1)} \sum_{d_2 \in D(G_2)} \sum_{t_1 \in ND(G_1)} \sum_{t_2 \in ND(G_2)} k(d_1, d_2, t_1, t_2),$$

where k is a positive definite kernel. In this paper we investigate the GSP kernel which considers node pairs equal if they have the same shortest distance *and* the same number of shortest paths. Which gives us the following definition, called the *generalized shortest path index* (GSPI) kernel [7].

$$K_{\mathrm{GSPI}}(G_1, G_2) = \sum_{d_1 \in D(G_1)} \sum_{d_2 \in D(G_2)} \sum_{t_1 \in ND(G_1)} \sum_{t_2 \in ND(G_2)} \mathbb{1}\left[d_1 = d_2\right] \mathbb{1}\left[t_1 = t_2\right].$$

It is easy to see that this is equivalent to the inner product of the GSPI feature vectors of G_1 and G_2.

Computing the SPI and GSPI feature vectors can be done efficiently. One possible method is to use Dijkstra's algorithm [6] for each node in a graph. Doing so takes $\mathcal{O}(nm + n^2 \log n)$ time for one graph and gives us the shortest path length of all node pairs in a graph. This information is exactly what is needed to construct a SPI feature vector. When running Dijkstra's algorithm, the algorithm actually computes all shortest paths between each node pair, although most applications do not store this information, by saving this information we can count the number of shortest paths between each node pair in the graph *without any extra cost in running time*. This means that calculating a GSPI feature vector takes basically the same time as calculating a SPI feature vector.

Note that all the graph kernels used in this paper are inner products between finite dimensional feature vectors. We choose to still call them graph kernels in order to preserve the connection to other researchers' work on graph kernels, see for instance [3], [4], [9].

5 Analysis

In this section we analyze in which situations, and why, the GSP kernel outperforms the SP kernel. Further evidence for our analysis can be found in Sect. 6, which contains our experimental results.

In this section, instead of considering the number of node *pairs* which are at distance d from each other, we consider the number of nodes at a particular distance from a fixed source node s. By $n_{d,x}^{(k)}$ we mean the number of nodes that are at distance d from s and have exactly x number of shortest paths to s, in a k-cluster graph.

5.1 Preliminary approximations

Here we give some preliminary approximations that we use in order to derive our main result. We estimate the expected number of nodes at distances 1 and 2, from a fixed node s, in a k-cluster graph. Our analysis closely parallels [7]. Consider the probability that s is connected to some other fixed node t. Let $F_{t,1}$ be the probability that s and t are connected. $F_{t,1}$ is obviously p if t is in the same cluster as s and q_1 otherwise. Let $F_{t,2}$ be the probability that there exists at least one path of length 2 between s and t. Let $A_{u,v}$ be the event that u and v are connected. $F_{t,2}$ is then equal to

$$F_{t,2} = \Pr\left[\bigvee_{v \in V \setminus \{s,t\}} A_{s,v} \wedge A_{v,t}\right]. \quad (1)$$

$F_{t,2}$ can be calculated using the inclusion-exclusion principle and will give a different result depending on if t is in the same cluster as s or not. Let $f_{t,2}$ denote the probability that s and t are at distance 2. This happens if s and t have a path of length 2 and no path of length 1. I.e. $f_{t,2} = F_{t,2} - F_{t,1}$. Note that $f_{t,2}, F_{t,2}$ and $F_{t,1}$ vary depending on where t is. Let f_2 denote the probabillity that s is at distance 2 to a *random* node t. f_2 is simply the weighted average sum over the cases when t is in the same cluster as s and when t is not in the same cluster as s. The probability of t being in the same cluster as s is simply $1/k$ and the probability that t is not in the same cluster as s is $(k-1)/k$. With this f_2, we are able to estimate that the expected number of nodes at distance 2 is $(n-1)f_2$.

In order to simplify (1), we use an approximation of the inclusion-exclusion principle from [7]. I.e. we approximate

$$\Pr\left[\bigcup_{i=1}^{l} E_i\right] \approx 1 - \exp\left(-\sum_{i=1}^{l} \Pr[E_i]\right).$$

Where E_i, in the case of (1), is $A_{s,v} \wedge A_{v,t}$.

5.2 Analysis of GSPI feature vectors

Here we analyze the expected GSPI feature vectors, $\mathrm{E}[\,\boldsymbol{v}_{\mathrm{gsp}}^{(k)}\,]$ and $\mathrm{E}[\,\boldsymbol{v}_{\mathrm{gsp}}^{(k+1)}\,]$, where the graphs are generated as specified in Sect. 3. Here we consider the number of shortest paths between a fixed source node s and a random target node t. We focus on the part of the GSPI feature vectors that corresponds to nodes at distance 2 from s, namely the subvectors $[\mathrm{E}[n_{2,x}^{(k)}]]_{x \geq 1}$ and $[\mathrm{E}[n_{2,x}^{(k+1)}]]_{x \geq 1}$. Since we in this section consider feature vectors for a fixed s, note that for instance, $\mathrm{E}[n_{2,4}^{(k)}]$ is the expected number of nodes, that are at distance 2 from s and have 4 number of shortest paths to s in a k-cluster graph.

We first consider a k-cluster graph but we use q instead of q_1 so that we are later able to replace q by either q_1 or q_2. Assume, without loss of generality, that s is in cluster 1. Note that s has an expected $(n/k)p$ number of nodes at distance 1, in cluster 1. s also has an expected $(n/k)q$ number of nodes at distance 1 in each of the other $k-1$ clusters. For our analysis, we assume that the number of nodes at distance 1, in each cluster, is actually equal to the expected number of such nodes. We also assume that each node at distance 1 is connected to any node at distance 2 *independently* at random. We give evidence later in this section that these are actually reasonable assumptions.

We now analyze the number of shortest paths between s and t, when s and t are at distance 2. We first consider the case when t is in cluster 1. In this case we know that s has an expected $(n/k)p$ nodes at distance 1 in cluster 1. Due to our assumption of independence of each of those $(n/k)p$ nodes having an edge to t, we say that each such

node is connected to t with probability p independently at random. Which means that if we consider the number of shortest paths between s and t, that goes through cluster 1, the number of shortest paths is distributed according to the binomial distribution $\mathrm{Bin}((n/k)p, p)$. There could also be shortest paths that go from s to cluster 2 then to t, from s to cluster 3 then to t and so on. For any of these situations, the number of shortest paths is distributed according to the distribution $\mathrm{Bin}((n/k)q, q)$. There are obviously $k-1$ such cases. The final distribution of the number of shortest paths when s and t are both in cluster 1 is then

$$\mathrm{Bin}(\frac{np}{k}, p) + (k-1)\mathrm{Bin}(\frac{nq}{k}, q). \quad (2)$$

Similarly, we can calculate that, when t is not in cluster 1, the number of shortest paths between s and t is distributed according to the distribution

$$\mathrm{Bin}(\frac{np}{k}, q) + \mathrm{Bin}(\frac{nq}{k}, p) + (k-2)\mathrm{Bin}(\frac{nq}{k}, q). \quad (3)$$

Note here that we have two different cases, the first case where t is in cluster 1 and the second case where t is in any other cluster. Since the position of t is randomly determined, the final distribution of the number of shortest paths between s and t is a *mixture distribution* [11] of (2) and (3). Where the weight of the first distribution is the number of nodes at distance 2 in cluster 1 over the total number of nodes at distance 2. Which we write as

$$\frac{n_2^{(1,k)}}{n_2^{(k)}} = w_1^{(k)}.$$

We call this weight $w_1^{(k)}$. The weight of the second distribution is the sum of the number of nodes at distance 2 in all clusters except cluster 1, over the total number of nodes at distance 2. Written as

$$\sum_{i=2}^{k} \frac{n_2^{(i,k)}}{n_2^{(k)}} = w_2^{(k)}.$$

We call this weight $w_2^{(k)}$. Note that $w_1^{(k)} + w_2^{(k)} = 1$. The final mixture distribution, for the number of shortest paths between s and a random node t, that is at distance 2 from s, is then

$$w_1^{(k)}(\mathrm{Bin}(\frac{np}{k}, p) + (k-1)\mathrm{Bin}(\frac{nq}{k}, q)) + \quad (4)$$
$$w_2^{(k)}(\mathrm{Bin}(\frac{np}{k}, q) + \mathrm{Bin}(\frac{nq}{k}, p) + (k-2)\mathrm{Bin}(\frac{nq}{k}, q)).$$

This mixture distribution must then have two peaks, the first one being

$$x_{\mathrm{peak}}^{(1,k)} = \frac{n}{k}(p^2 + (k-1)q^2)$$

and the second one being

$$x_{\mathrm{peak}}^{(2,k)} = \frac{n}{k}(2pq + (k-2)q^2).$$

When using the above formulas for a k-cluster graph, we simply replace q by q_1. To obtain the relevant formulas for a $k+1$-cluster graph, we need to replace q by q_2 and k by $k+1$.

We denote the difference between the peaks for a k-cluster graph as

$$\Delta x_{\mathrm{peak}}^{(k)} = x_{\mathrm{peak}}^{(1,k)} - x_{\mathrm{peak}}^{(2,k)} = \frac{np^2\beta^2}{k}. \quad (5)$$

Note that the two weights in (4) will be different for the k-cluster graphs and the $k+1$-cluster graphs. In particular, for fixed values of p, β and k, we always have $w_2^{(k)} < w_2^{(k+1)}$, since the weight of the second distribution increases when the number of clusters increases (since then there is a lower chance of t being in cluster 1). Also note that $w_1^{(k)}$ decreases when k grows. If $w_1^{(k)}$ becomes negligible, we will not have a two peak shape even if the two peaks are separated by a wide margin. This is however never the case for the parameters we consider in our experiments, i.e. $k \in [2, 9]$.

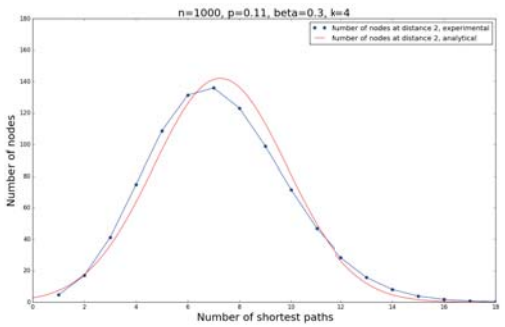


Fig. 1. Comparison between the average experimental and analytical feature subvectors $[\mathrm{E}[n_{2,x}^{(k)}]]_{x\geq 1}$. For n=1000, p=0.11, beta=0.3, k=4.

In the above semi-theoretical analysis, we made two noteworthy assumptions. First that the number of nodes at distance 1 from a fixed node s, is equal to its expected value. We also assumed that all of the distance 1 nodes are connected to distances 2 nodes independently at random, which finally gave us the distribution (4) for the number of shortest paths between s and a random node t at distance 2 from s. In order to provide some evidence that this model is actually reasonably close to what really happens, we provide two figures where we compare the subvector $[\mathrm{E}[n_{2,x}^{(k)}]]_{x\geq 1}$ from the experiments and the subvector which we obtain by using (4) multiplied by the number of nodes at distance 2. We use the approximation from Sect. 5.1 to approximate the number of nodes at distance 2 and substitute the binomial distributions in (4) for normal distributions in order to simplify summing the distributions. The results can be seen in Figures 1 and 2. As can be seen, the distributions obtained from our analysis are very close to the experimental values, no matter if the distribution appears to have only one peak (the two peaks are very close), or if the two peaks are far away.

It is important to note that if the difference between the peaks is large enough, the distribution will look very different compared to when the difference between the peaks is close to 0. If the difference between the peaks is large, the distribution will look skewed compared to a standard normal distribution. This can be seen in Fig. 3, which shows the average subvectors $[\mathrm{E}[n_{2,x}^{(k)}]]_{x\geq 1}$, when $n = 1000$, $p = 0.15$, $k = 2$ for $\beta = 0.2$ and

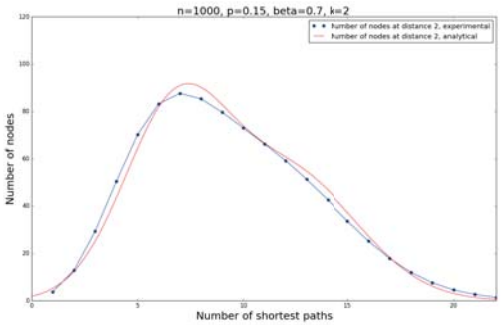


Fig. 2. Comparison between the average experimental and analytical feature subvectors $[\mathrm{E}[n_{2,x}^{(k)}]]_{x\geq 1}$. For n=1000, p=0.15, beta=0.7, k=2.

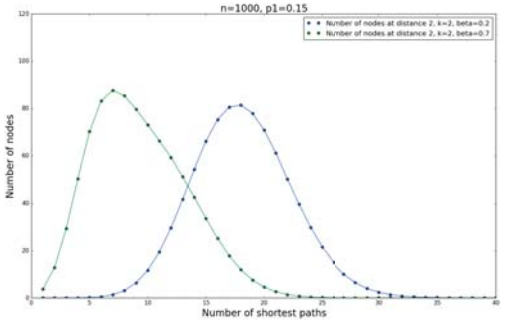


Fig. 3. Average subvectors $[\mathrm{E}[n_{2,x}^{(k)}]]_{x\geq 1}$, when $n = 1000$, $p = 0.15$, $k = 2$ for $\beta = 0.2$ and $\beta = 0.7$. The difference between the peaks when $\beta = 0.2$ is 0.45 and the difference between the peaks when $\beta = 0.7$ is 5.51.

$\beta = 0.7$. Note that the difference between the peaks is a lot greater when $\beta = 0.7$ than when $\beta = 0.2$. In fact the difference between the two peaks is equal to 0.45 when $\beta = 0.2$ and 5.51 when $\beta = 0.7$. The distributions of the k and $k+1$-cluster graphs also appears to look more different when the difference between the peaks is large. Figure 4, contains the average subvectors $[\mathrm{E}[n_{2,x}^{(k)}]]_{x\geq 1}$ and $[\mathrm{E}[n_{2,x}^{(k+1)}]]_{x\geq 1}$, when $n = 1000$, $p = 0.15$, $k = 2$, $\beta = 0.7$. The difference between the peaks for the k-cluster graphs is 5.51 and the difference between the peaks for the $k+1$-cluster graphs is 2.07. If the distance between the peaks is low, the feature vectors tend to look the same for k-cluster graphs and $k+1$-cluster graphs. An example of this can be seen in Fig. 5. Obviously, for such datasets, it is not possible to distinguish the feature subvectors.

Now we propose a "hidden parameter", a major factor in determining the accuracy of the GSPI based SVM classifier. It is well known that the difficulty of distinguishing between a simple one peak distribution and a mixture distribution with two peaks, is related to the ratio of the distance of the two peaks and the standard deviation of the distributions [11]. Here we propose to use this ratio for the hidden parameter. Note that the standard deviation of both (2) and (3) can be approximated as $\sqrt{nq_1^2(1-q_1)}$. Thus, the ratio can be approximated by

$$R_{n,p,k,\beta} = \frac{np^2\beta^2}{k}/\sqrt{nq_1^2(1-q_1)} \approx \sqrt{n}\ p\ \beta^2/(k(1-\beta)). \quad (6)$$

Which we conjecture to be the major factor determining the accuracy of the GSPI based SVM, for the graph classification problem of deciding if a graph contains k or k+1 clusters. In Sect. 6.3 we derive some evidence supporting this conjecture based on experimental results.

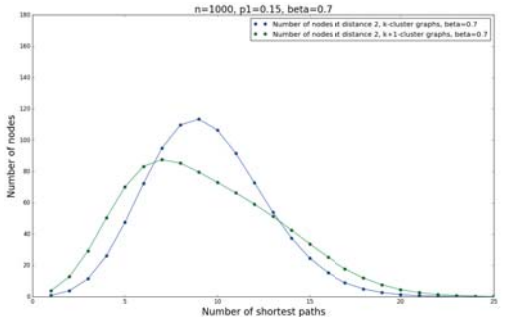


Fig. 4. Average subvectors $[\mathrm{E}[n_{2,x}^{(k)}]]_{x\geq 1}$ and $[\mathrm{E}[n_{2,x}^{(k+1)}]]_{x\geq 1}$, when $n = 1000$, $p = 0.15$, $\beta = 0.7$, $k = 2$. The difference between the peaks for the k-cluster graphs is 5.51 and the difference between the peaks for the $k+1$-cluster graphs is 2.07.

6 Experiments

In this section we show and explain experimental results.

6.1 Dataset specifications and experiment parameters

All datasets are generated according to the models $G(n,k,p,q_1)$ and $G(n,k+1,p,q_2)$. Where each dataset consists of 200 graphs from each model (400 in total). We performed the experiments for the parameters $n = 1000$, $p \in \{0.07, 0.09, 0.11, 0.13, 0.15\}$, $k \in \{2, 3, \ldots, 9\}$ and $\beta \in \{0.2, 0.22, 0.24, \ldots, 0.7\}$. For the SVM, we used *libsvm* [5], with 10-fold cross validation. For the cross validation we tried $C = 2^i$, for $i = [0, 49]$ and the result from the best C was used as the accuracy. Each experiment was repeated 10 times, with different random seeds, and the final accuracy is the average of the 10 runs. If any of the 10 runs resulted in less than 56% accuracy, the final accuracy for that dataset and kernel was set to 56% in order to save running time. Preliminary

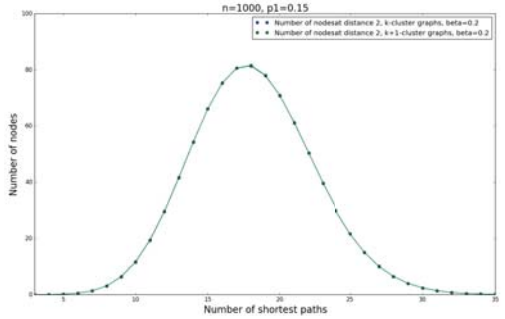


Fig. 5. Average subvectors $[\mathrm{E}[n_{2,x}^{(k)}]]_{x\geq 1}$ and $[\mathrm{E}[n_{2,x}^{(k+1)}]]_{x\geq 1}$, when $n = 1000$, $p = 0.15$, $\beta = 0.2$, $k = 2$. The difference between the peaks for the k-cluster graphs is 0.45 and the difference between the peaks for the $k+1$-cluster graphs is 0.17. The distributions in this picture are indistinguishable.

TABLE 1
COMPARISON OF ACCURACY FOR THE SPI KERNEL AND THE GSPI KERNEL. $p = 0.15$, $k = 2$.

β	SPI accuracy	GSPI accuracy
0.2	56%	63.6%
0.22	56%	70.2%
0.24	56%	78.2%
0.26	56%	85.3%
0.28	56%	92.0%
0.3	56%	95.2%
0.32	56%	98.4%

TABLE 2
COMPARISON OF ACCURACY FOR THE SPI KERNEL AND THE GSPI KERNEL. $p = 0.09$, $k = 2$.

β	SPI accuracy	GSPI accuracy
0.26	60.3%	71.3%
0.28	61.7%	74.7%
0.3	60.3%	82.3%
0.32	62.2%	84.1%
0.34	68.1%	91.9%
0.36	74.5%	92.8%
0.38	78.9%	98.5%

TABLE 3
COMPARISON OF ACCURACY FOR THE SPI KERNEL AND THE GSPI KERNEL. $p = 0.15$, $k = 4$.

β	SPI accuracy	GSPI accuracy
0.26	56%	64.4%
0.28	56%	68.0%
0.30	56%	73.0%
0.32	56%	87.2%
0.34	56%	92.1%
0.36	56%	94.4%
0.38	56%	97.8%

TABLE 4
COMPARISON OF ACCURACY FOR THE SPI KERNEL AND THE GSPI KERNEL. $p = 0.09$, $k = 4$.

β	SPI accuracy	GSPI accuracy
0.34	57.9%	64.2%
0.36	60.8%	74.7%
0.38	64.2%	80.0%
0.4	70.0%	86.2%
0.42	73.6%	90.3%
0.44	78.9%	92.5%
0.46	84.9%	97.1%

experiments showed that $i < 1$ never gave any competitive accuracy. We used the SPI and GSPI kernels, with their feature vectors as defined in Sect. 4, with the modification that each feature vector is normalized by its Euclidean norm, this means that all inner products are in $[0, 1]$.

6.2 Results

Both kernels increase in accuracy when β increases. The SPI does not perform good when p is very large, this is because of the fact that for large values of p, all node pairs are at very close distances. For none of the datasets did the SPI kernel significantly outperform the GSPI kernel, although for very low values of p, the kernels perform similarly in terms of accuracy. For larger values of p, the GSPI significantly outperforms the SPI kernel. A comparison between the accuracies of the kernels can be seen in Tables 1, 2, 3 and 4.

The SPI kernel is only able to detect changes in the number of nodes at certain distances, i.e. $n_1, n_2 \ldots$. Also note that the expected number of edges is the same for the k-cluster graphs and the $k+1$-cluster graphs, which means $\mathrm{E}[n_1^{(k)}] = \mathrm{E}[n_1^{(k+1)}]$. Because of this, for datasets where all nodes are at distances 1 and 2, the SPI kernel will not be able to gain any advantage over a random classifier. The number of nodes at distances 3 and greater, is larger for smaller values of p and larger values of β.

6.3 The accuracy of the GSPI kernel

We plot the accuracy of the GSPI kernel for $p = 0.11$ and $p = 0.15$. These results can be seen in Figures 6 and 7. Our analysis is focused on the subvectors at distance 2 for analyzing the accuracy of the GSPI kernel. The reason for this is because of the fact that for most of the datasets, the number of node pairs that are at distances 1 and 2 is over 50%, meaning that the number of such node pairs is never negligible. In fact, for certain datasets, nearly all the node pairs are at distances 1 and 2. Such a range of parameters is for instance, $p = 0.15, \beta \in [0.2, 0.4], k = 2, 3$, where the number of node pairs at distance more than 2 is always less than 0.1% of the total number of node pairs. This fact, that the number of node pairs at distances greater than 2 is very small, is one reason that the SPI kernel does not perform well for those datasets.

Now let us derive some evidence from our experimental results supporting our conjecture, that is, the ratio $R_{n,p,k,\beta}$ is the major factor determing the accuracy of the GSPI based SVM classifier. Unfortunately, it is computationally hard to get data for various graph sizes n. Thus, in this paper, we consider one reasonably large graph size, namely, $n = 1000$, fixed, and obtain experimental results by changing the other parameters, p, k, and β. Thus, in the following discussion, we regard n in the ratio $R_{n,p,k,\beta}$ as a constant.

First we note the relation between β and the accuracy of the GSPI kernel; see, Figures 6 and 7. From the figures

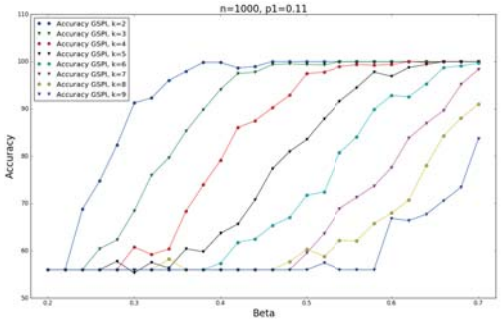


Fig. 6. Accuracy of the GSPI kernel when $p = 0.11$.

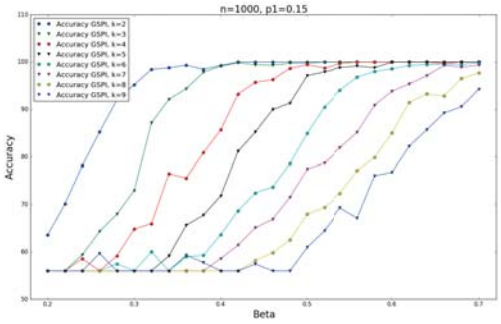


Fig. 7. Accuracy of the GSPI kernel when $p = 0.15$.

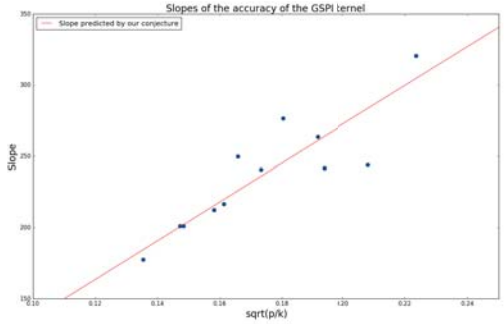


Fig. 8. The slopes of the accuracy of the GSPI kernel for $p \in \{0.11, 0.13, 0.15\}$, $k \in \{3, 4, 5, 6\}$ and the slope as predicted by our conjecture that the slope is proportional to $\sqrt{p/k}$.

it seems that the accuracy increases linearly in β for the interval of β where the accuracy has started to increase above 60% until it reaches close to 100%. That is, for such an interval of β, the accuracy could be written as $a\beta+b$ at least approximately. Note on the other hand that

$$\sqrt{R_{n,p,k,\beta}} \propto \left(\sqrt{p/k}\right) \beta/\sqrt{1-\beta},$$

and that $\beta/\sqrt{1-\beta}$ is close to linear for that interval of β. Thus, if $R_{n,p,k,\beta}$ were the major factor of the accuracy, the accuracy can be expressed as

$$\sqrt{R_{n,p,k,\beta}} + b \approx c\left(\sqrt{p/k}\right) \beta + b,$$

and hence, the slopes of graphs, e.g., in Figures 6 and 7 (i.e., how quickly the accuracy increases when β increases) should be proportional to $\sqrt{p/k}$. We can check this point from our experimental results.

More specifically, we check whether the slopes are proportional to $\sqrt{p/k}$ as follows. First calculate the slope, from the experiments, of the accuracy of the GSPI kernel for several combinations of the parameters p and k. The slopes were obtained by picking the intervals of β values for which the GSPI accuracy seems to increase linearly, where we considered $p \in \{0.11, 0.13, 0.15\}$, $k \in \{3, 4, 5, 6\}$. We then fitted a line $a\beta+b$ using the least squares method in order to obtain the slope a. We then plotted the obtained slopes (values of a) against $\sqrt{p/k}$ and fitted it by a function $g(x) = cx$, again using the least squares method. The result of this can be seen in Fig. 8, where c was determined to be 1362.8. In the figure, each dot represents an experimental slope value obtained from one combination of p and k and the line is the slope that we predict based on

$$\text{Slope} \propto \sqrt{p/k}.$$

As can be seen in the figure, the slope obtained from our conjecture is reasonably close to the real values.

7 Conclusions

We have provided a semi-theoretical analysis of the feature vectors of the GSPI kernel for a random graph classification problem where we classify graphs as having k or $k+1$ number of clusters. Our experiments showed that the analysis closely parallels what really happens. We conjectured that the important parameter for determining the accuracy of the GSPI kernel is related to the distance between the peaks of the mixture distribution of number of shortest paths of distance 2, divided by the standard deviation of the two distributions forming the mixture distribution. We came up with a conjecture for determining what the slope of the accuracy of the GSPI kernel will be, and verified for certain datasets that our conjectured slope value is close to the real value.

Future work should focus on extending the analysis to other values of n. Throughout the paper we only considered $n = 1000$, it would be interesting to analyze what behavior the GSPI kernel has as n increases.

References

[1] C. Bilgin, C. Demir, C. Nagi, and B. Yener. Cell-graph mining for breast tissue modeling and classification. In *Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE*, pages 5311–5314. IEEE, 2007.

[2] B. Bollobás. *Random graphs*. Springer, 1998.

[3] K. M. Borgwardt and H.-P. Kriegel. Shortest-path kernels on graphs. In *Prof. of ICDM*, 2005.

[4] K. M. Borgwardt, C. S. Ong, S. Schönauer, S. Vishwanathan, A. J. Smola, and H.-P. Kriegel. Protein function prediction via graph kernels. *Bioinformatics*, 21(suppl 1):i47–i56, 2005.

[5] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[6] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

[7] L. Hermansson, F. D. Johansson, and O. Watanabe. Generalized shortest path kernel on graphs. In *Discovery Science*, pages 78–85. Springer, 2015.

[8] L. Hermansson, T. Kerola, F. Johansson, V. Jethava, and D. Dubhashi. Entity disambiguation in anonymized graphs using graph kernels. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 1037–1046. ACM, 2013.

[9] F. Johansson, V. Jethava, D. Dubhashi, and C. Bhattacharyya. Global graph kernels using geometric embeddings. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 694–702, 2014.

[10] S. D. A. Kolla and K. Koiliaris. Spectra of random graphs with planted partitions.

[11] M. F. Schilling, A. E. Watkins, and W. Watkins. Is human height bimodal? *The American Statistician*, 56(3):223–229, 2002.

Combining Ensembles

Ulf Johansson*, Henrik Linusson†, Tuve Löfström†, Cecilia Sönströd†
*Department of Computer Science and Informatics, Jönköping University, Sweden
Email: ulf.johansson@ju.se
†Department of Information Technology
University of Borås, Sweden
Email: {henrik.linusson, tuve.lofstrom, cecilia.sonstrod}@hb.se

***Abstract*—Ensemble techniques are the standard choice for obtaining robust and high predictive performance. Two of the most popular ensemble techniques are random forests and bagged neural networks. Although often having similar accuracy, it is observed that a neural network ensemble and a random forest often disagree, i.e., produce different predictions, on a substantial number of instances, thus indicating an opportunity to further improve predictive performance by somehow actively selecting one of the ensembles for each prediction. In this study, several straightforward ways of utilizing two different ensembles are suggested and empirically evaluated, using 24 publicly available data sets. The results show that the simple method of selecting the ensemble which is most certain on its prediction achieved the best predictive performance, clearly outperforming both underlying ensembles.**

***Keywords*—Predictive modeling, Ensembles, Bagging, Random forest, Neural networks**

I. Introduction.

In the data mining task predictive modeling, the aim is to build a model capturing the relationship between an input vector $\mathbf{x}$ and a target concept y; in effect approximating the function $f(\mathbf{x}, y)$. When the target concept is a continuous variable, the task is called *regression*, while *classification* concerns categorical target variables, taking their values from a pre-determined set of class labels. For predictive models, the overriding concern is predictive performance, i.e., how well the model predicts the target value based on an input vector, for previously unseen examples. For classification, the simplest measure for this property is *accuracy*, which states the percentage of correct predictions on novel data instances. A wealth of machine learning techniques for producing classification models exist, ranging from transparent rule or decision tree models built using relatively simple algorithms, to opaque models such as artificial neural networks (ANNs) or support vector machines. Generally, opaque predictive models achieve better predictive performance than transparent models. It is firmly established in machine learning research (see e.g., [1] or [2]), that predictive performance can be further improved by combining several (different) models into an *ensemble*. In an ensemble, multiple *base models*, for classification called *base classifiers*, are combined into a composite model, where the ensemble prediction is a function of all included base model predictions. The basic idea behind this scheme is that a collection of models, which make their errors on different instances, will perform better than any single one of the models, since aggregating several models, using averaging or majority voting, will eliminate uncorrelated base model errors; see e.g. [3]. Based on their robustly high predictive performance, ensembles models, such as *random forests* [4] constitute the state-of-the-art in predictive modeling and are available in most commercial data analysis packages. Studying how errors are distributed between base classifiers in an ensemble, it becomes apparent that some instances are intrinsically difficult to predict, i.e., almost all base classifiers in the ensemble get them wrong. However, for a substantial proportion of ensemble errors, a number of base classifiers actually make a correct prediction. This reasoning can also be applied when considering an arbitrary pair of predictive models; the fact that they have similar predictive performance does not entail that they make their errors on the same instances. This implies that there is a possibility to obtain a correct prediction, if one could only choose which model to trust. Obviously, for this to be workable in practice, model predictions must be combined automatically to produce a single prediction for each instance.

The purpose of this study is to conduct a thorough empirical investigation of how the above observation, regarding differences on the instance level, between two models with similar predictive performance, can be exploited to increase overall predictive performance. In short, two different ensemble classifiers, both highly accurate and with similar predictive performance over a large number of data sets, are built using off-the-shelf techniques. Based on observations of model characteristics regarding errors and error distributions, different ways of combining ensemble predictions are empirically evaluated over a large number of data sets.

II. Background.

Simply put, an ensemble is constructed by training a set of L base classifiers and combining their predictions. When performing classification, the predictions of the base classifiers can be combined using, e.g., majority voting. For the ensemble to be effective, the base classifiers must commit their errors on different instances, which is the informal meaning of the term *diversity*. Arguably the simplest and most intuitive diversity measure, *disagreement* calculates the disagreement between all pairs of classifiers as the ratio between the number of instances on which one classifier is correct and the other incorrect and the total number of instances [5]:

$$\mathrm{D}_{j,k} = \frac{N^{01} + N^{10}}{N} \quad (1)$$

This work was supported by the Swedish Foundation for Strategic Research through the project High-Performance Data Mining for Drug Effect Detection (IIS11-0053) and the Knowledge Foundation through the project Data Analytics for Research and Development (20150185).

where N^{01} and N^{10} measure the number of instances on which the two models j and k are correct (1) and incorrect (0). N is the total number of instances. The disagreement measure is the average over all pairs:

$$Dis = \frac{2}{L(L-1)} \sum_{j=1}^{L-1} \sum_{k=j+1}^{L} D_{j,k} \tag{2}$$

Brown and Kuncheva showed that the ensemble error when using majority voting can be decomposed into the average base classifier error and diversity [6]. The diversity of a base classifier $h_l(x) \in \{1, -1\}$ is, for this decomposition, defined as the disagreement between the classifier and the ensemble prediction $H(x) \in \{1, -1\}$: $d_l(x) = \frac{1}{2}(1 - h_l(x)H(x))$.

The decomposition of the ensemble error e_{maj} into the average error of the base classifiers e_{ind} and the diversity d_l for an individual instance is:

$$e_{\text{maj}}(x) = e_{\text{ind}}(x) - yH(x)\frac{1}{L}\sum_{l=1}^{L} d_l(x) \tag{3}$$

where $y \in Y = \{1, -1\}$. It is evident that the ensemble error is composed of the average base classifier error and a combination of the true target value, the ensemble prediction and the average diversity among the base classifiers. The major difference between the decomposition for majority voting and a similar decomposition for regression ensembles [7] is the inclusion of the true target value in equation (3).

Equation (3) calculates the majority vote error of a single instance. To calculate the majority vote error over all instances, E_{maj}, taking advantage of the fact that $yH(x) = 1$ when correct and $yH(x) = -1$ when incorrect, the integration with respect to the probability density function becomes

$$\begin{aligned} E_{\text{maj}} &= \int_x e_{\text{ind}}(x) - \int_x yH(x)\frac{1}{L}\sum_{l=1}^{L} d_l(x) \\ &= \int_x e_{\text{ind}}(x) \\ &\quad - \underbrace{\int_{x+} \frac{1}{L}\sum_{l=1}^{L} d_l(x)}_{\text{good diversity}} \\ &\quad + \underbrace{\int_{x-} \frac{1}{L}\sum_{l=1}^{L} d_l(x)}_{\text{bad diversity}} \end{aligned} \tag{4}$$

where $x+$ refers to instances on which the ensemble is correct and $x-$ refers to instances on which the ensemble is wrong. What the decomposition essentially shows is that increasing the disagreement is beneficial for instances where the ensemble is correct and detrimental for instances where the ensemble is wrong, hence the labels 'good' and 'bad' diversity.

Brown et al. [8] introduced a taxonomy of methods for creating diversity. The most important distinction made is between explicit methods, where some metric of diversity is directly optimized, and implicit methods, where the method is likely to produce diversity without actually targeting it. A common procedure in many explicit methods is to first train a set of classifiers and then select, based on some diversity metric, a subset of classifiers to combine when predicting [9]. The selection can either be static, when the same subset is always used, or dynamic, when a new subset is selected for each new instance. However, several studies have shown that diversity is not likely to be useful as the optimization criterion for this procedure [10]–[12].

Bagging [13] is an implicit method which achieves diversity by training each base classifier using different emulated training sets obtained using resampling with replacement. Each training set (called a *bootstrap*) generally consists of the same number of instances as the entire data set available for training, with the result that approximately 63.2 % of available instances are included in each bootstrap. Since only a subset of the instances are used to train each classifier, there is always a proportion of the available instances, called *out-of-bag* (OOB) instances, not used to train a classifier. By forming an ensemble composed of only the classifiers for which an instance is out-of-bag, it is possible to get an unbiased estimate of the bagging ensemble's performance, while still using all instances for training. Since the out-of-bag ensemble is approximately $1/3$ of the size of the entire bagging ensemble, the out-of-bag error estimate tends to overestimate the actual error made by an ensemble, simply because a larger ensemble is normally a stronger model.

Implicitly targeting diversity is inherent in random forests. Random forests consist of decision trees and achieve their diversity by introducing randomness in both instance and feature selection; bagging is used to select which instances to use when training each tree and each split is based on only a randomized subset of the attributes. Similarly to standard bagging, out-of-bag estimates can be used to estimate the predictive performance of a random forest.

Diversity can also be implicitly targeted when constructing ensembles of neural networks. In an evaluation of different ways of implicitly creating diversity in ensembles of neural networks [14], it was shown that bagging is a reliable method for consistently achieving diversity.

Since ensembles of neural networks and random forests are both state-of-the-art techniques with high predictive performance, it might be expected that two of these ensemble models would agree to a large degree in their predictions. To explore how they perform in relation to each other, Figs. 1-3 show the proportion of base classifiers correctly predicting each instance, for a sample data set. In Fig. 1, all instances are plotted for both random forests and ensembles of neural networks, sorted on the proportion of base classifiers being correct. In Fig. 2, only instances on which the opposing ensemble is correct are plotted. This means that only instances on which the random forest (RF) is correct are plotted in the plot with the ensemble of neural networks (Ens) and vice versa. Fig. 2 plots only instances on which a majority of the base classifiers of the opposing ensemble are incorrect.

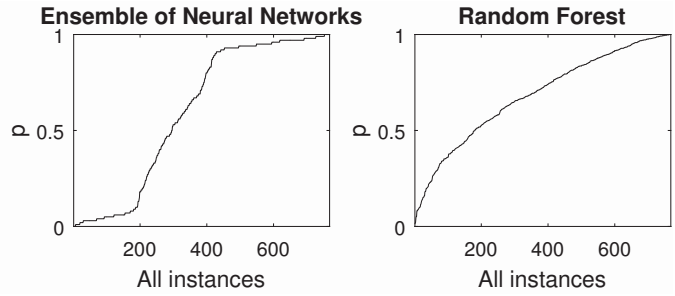


Fig. 1. Proportions of models correctly predicting instances on the Liver data set. The left plot shows the proportions for the ANN-ensemble and the right plot for the random forest. Instances are sorted on the proportion of the models being correct.

It is evident from Fig 1 that random forests and ensembles of neural networks, while being accurate to approximately the same degree, distribute their base classifiers errors quite differently. The neural networks tend, to a much larger degree than the trees in the random forests, to agree in their predictions. In other words, the neural networks are less diverse but more accurate than the trees, on average. The question remains, however, if the models agree on individual instances.

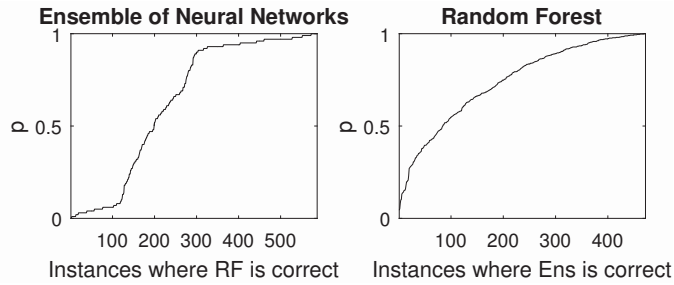


Fig. 2. The same setup as in Fig. 1 with the difference that only instances correctly predicted by the opposing ensemble are included.

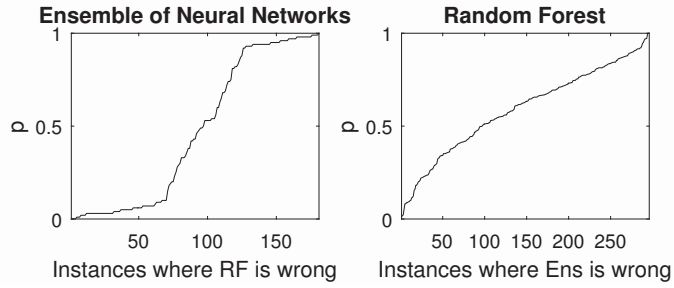


Fig. 3. The same setup as in Fig. 1 with the difference that only instances incorrectly predicted by the opposing ensemble are included.

As is clearly seen in Figs. 2 and 3, the proportion of instances on which these two highly accurate ensemble models disagree is fairly large. As expected, there are obviously many instances on which both models agree but there are also many instances on which they strongly disagree. The neural networks in particular strongly agree on the correct prediction on many instances that the random forest has failed to predict correctly. The decision trees in the random forest, on the other hand, agree to a much lower degree on the instances the ensemble of neural networks had failed to predict correctly. See Table I for results on the performance of individual models on all data sets.

A. Related work.

The goal of finding the most suitable learning algorithm, be it in general, with regard to a specific decision problem, or with regard to specific subspaces of a certain problem domain, is central to the predictive modeling task. Early approaches, still commonly used today, include the cross-validation [15], [16], generalized cross-validation [17] and bootstrapping [15] methods. These winner-takes-all approaches allow an analyst to select, from a pool of candidate algorithms, the most accurate one, either for a specific classification problem or for a set of classification problems.

The idea of stacked generalization [18]–[20] builds on the cross-validation method, but, rather than selecting the single most accurate learning algorithm, a composite ensemble model is created using a combination function. The goal of stacking learners is, of course, to create a composite model that is more accurate than the models created by any of the individual candidate learning algorithms. Closely related to the idea of stacking is the field of classifier fusion, in which heterogeneous ensemble models (containing models induced from different learning algorithms) are found through different combination methods [21]–[25].

Another related field of study is the overproduce-and-select scheme sometimes used in ensemble learning [9]. Here, a large pool of learners is first created, either by inducing, with randomness, multiple models using the same learning algorithm, e.g., [26], or by utilizing multiple learning algorithms, e.g., [24] (the overproduction step). Second, rather than utilizing the entire pool of learners as an ensemble model (using, e.g., voting), a subset of the candidate models are selected, using some heuristic, in order to attempt to construct a sub-ensemble more accurate than the model represented by the full model pool (the selection step).

All methods described thus far are *static*, i.e., once a certain model (ensemble or single model) has been selected, that model is used to predict the outputs of all future test patterns. With the dynamic overproduce-and-select scheme [27] (often called *dynamic ensemble selection*), sub-ensembles are selected on-line to predict the output of each test pattern, typically based on some characteristics of the particular test pattern or the (sub-)ensemble predictions. Examples include estimating and optimizing sub-ensemble performance locally in an explicit manner, i.e., by assessing the performance of ensemble members using the nearest-neighbor rule (based on performance on training or validation data) and creating an ensemble from locally well-performing models [28]–[30], as well as selecting sub-ensembles that are confident in their predictions, i.e., selecting sub-ensembles where there is little discrepancy amongst the voting members [31].

III. Method.

As described in the introduction, the overall purpose is to investigate whether it is possible to increase predictive performance by using two separate ensembles with different inductive biases. As an alternative to the plethora of highly

technical and very specialized ensemble schemes, we in this paper evaluate only straightforward approaches.

For this purpose, we start out with one ensemble of bagged neural nets and one random forest. For the neural network ensemble, we use 100 base classifiers, while the random forest has 500 trees. We believe these to be reasonable choices, specifically because the ensembles are large enough to allow for accurate out-of-bag estimates. In addition, adding more models would probably only lead to marginal improvements, making it interesting to look for other solutions; in other words, had we used significantly smaller ensembles, the most obvious method to increase accuracy would have been to simply add more base classifiers.

In the experimentation, a total of eight different setups are evaluated:

- **ANN**: The ensemble consisting of 100 bagged ANNs.
- **RF**: The random forest with 500 trees.
- **Prob**: The probability estimates of each model are used in this setup. When classifying a test instance, the ensemble with the highest probability estimate is used, i.e., if the two ensembles disagree, the one that is most certain is used. This is in fact very similar to averaging all models of both ensembles, but using a weighting of 5 for each ANN base classifier.
- **Rank**: This setup is identical to *Prob*, with one exception. Since ANN ensembles tend to be less diverse, there is a risk that *Prob* might favor the ANN ensemble. With this in mind, this setup instead first ranks all test instances, according to how certain the ensemble is, based on the probability estimates. This ranking gives the lowest rank to the highest probability estimate, and it is performed individually for both ensembles. When predicting a test instance, the ensemble with the lowest rank for this particular test instance is used, with the motivation that the instance is considered to be relatively easier for that ensemble. It must be noted that this setup requires a set of test instances, i.e., it can not be used on individual, e.g., streaming, test instances.
- **OOB-g**: Here the ensemble with the highest out-of-bag accuracy on the training set (on each fold) is used for predicting all test instances.
- **OOB-l**: In this setup, which ensemble to use is determined for each test instance individually. More specifically, the out-of-bag error over the ten neighboring instances is calculated, and the ensemble with the lowest error is used for predicting the test instance. If there is a tie, the technique with the lowest global out-of-bag error is used.
- **Gated**: Here, an additional feed-forward ANN (one hidden layer with ten units) is trained to learn the difference in out-of-bag errors for the two ensembles, using the training instances as inputs. When predicting a test instance, this additional ANN is used to determine which of the two ensembles that should be used. Naturally, if the ANN predicts the random forest to be the most accurate, the random forest is used for that test instance, and vice versa.
- **Tie-breaker**: In this setup too, an additional ANN is used. But here, the ANN is trained on only those training instances where the two ensembles disagree in their out-of-bag predictions. When predicting a test instance, this additional model is used for the actual prediction when the two ensembles output different classes.

All experimentation was performed in MatLab, in particular using the Neural network and the Statistics tool boxes. For the ensemble models, the functions *patternnet* and *treebagger* were used for neural networks and random forests, respectively. All ANNs had one hidden layer and, in order to introduce some additional diversity, the number of hidden units was, for each ANN, randomized from the uniform distribution $[\lfloor \frac{a}{2} \rfloor, a]$, where a is the number of attributes. As mentioned above, the number of random trees was set to 500, and the number of ANNs trained was 100. For the network training, resilient backpropagation (*rprop* in MatLab) was used. All other parameter values were left at their default values.

In all 24 publicly available data sets are used in the experimentation. All data sets are from the UCI [32] or KEEL [33] repositories. For the evaluation, predictive performance is measured using both accuracy and area-under-the-ROC-curve (AUC). While accuracy is based only on the classifications, AUC measures the ability to rank instances according to how likely they are to belong to a certain class; see e.g., [34]. AUC can be interpreted as the probability of ranking a true positive instance ahead of a false positive; see [35]. Since standard 10-fold cross-validation was used, all reported performance measures are averaged over the 10 folds.

IV. Results.

Table I below shows the results for the two ensembles. Looking first at ensemble accuracies, the mean results, when averaged over all data sets are very similar. In a head-to-head comparison, the random forest wins 13 data sets and the bagged ANNs 9. We see that individual neural networks are substantially more accurate than the corresponding random trees but, on the other hand, that diversity, measured as disagreement (*Dis*), is much higher in the random forest. The last column shows the proportion of all test instances where the two ensembles predict different class labels. Interestingly enough, the two ensembles disagree on almost 10% of all instances. For individual data sets, this number can be much higher, sometimes 20% or more.

TABLE I. ENSEMBLE RESULTS

	ANN			RF			
	ensAcc	bAcc	Dis	ensAcc	bAcc	Dis	Diff
colic	.832	.787	.165	.846	.723	.308	.059
creditA	.861	.845	.101	.880	.780	.243	.057
diabetes	.772	.758	.120	.764	.687	.310	.112
german	.691	.663	.209	.667	.609	.355	.108
haberman	.710	.721	.097	.686	.650	.291	.223
heartC	.841	.800	.149	.821	.732	.281	.066
heartH	.850	.811	.138	.823	.752	.245	.068
heartS	.826	.788	.150	.826	.726	.286	.074
hepati	.845	.810	.143	.845	.781	.235	.090
iono	.909	.885	.087	.934	.871	.156	.066
je4042	.722	.677	.242	.756	.675	.296	.204
je4243	.631	.595	.270	.656	.598	.368	.273
kc1	.758	.751	.060	.765	.673	.306	.104
kc2	.810	.787	.093	.789	.731	.245	.103
liver	.674	.609	.316	.733	.609	.395	.199
mw	.923	.914	.044	.910	.872	.119	.029
pc1req	.731	.640	.323	.683	.624	.326	.144
pc4	.900	.890	.047	.901	.854	.164	.050
sonar	.813	.771	.198	.832	.687	.369	.163
spect	.881	.844	.119	.885	.803	.199	.005
spectf	.824	.786	.164	.813	.733	.285	.094
ttt	.983	.976	.023	.991	.831	.254	.009
wbc	.955	.939	.039	.961	.910	.099	.019
vote	.867	.836	.139	.874	.799	.211	.054
Mean	**.815**	**.785**	**.140**	**.816**	**.735**	**.266**	**.099**

Table II below shows the accuracy results for the suggested ways of combining ensemble predictions, with ensemble accuracies from Table I repeated for comparison.

TABLE II. ACCURACY RESULTS FOR ENSEMBLE COMBINATIONS

	ANN	RF	Prob	Rank	OOB-g	OOB-l	Gated	Tie-br.
colic	.832	.846	.838	.838	.829	.818	.843	.832
creditA	.861	.880	.870	.867	.875	.878	.872	.867
diabetes	.772	.764	.777	.773	.770	.773	.764	.766
german	.691	.667	.679	.677	.691	.689	.663	.698
haberman	.710	.686	.707	.703	.710	.689	.700	.703
heartC	.841	.821	.848	.844	.838	.838	.841	.834
heartH	.850	.823	.840	.840	.829	.826	.843	.843
heartS	.826	.826	.822	.837	.807	.811	.819	.822
hepati	.845	.845	.858	.845	.839	.826	.832	.858
iono	.909	.934	.934	.934	.934	.929	.931	.903
je4042	.722	.756	.744	.744	.756	.733	.756	.722
je4243	.631	.656	.672	.672	.656	.669	.653	.661
kc1	.758	.765	.768	.770	.758	.758	.763	.755
kc2	.810	.789	.808	.802	.808	.797	.786	.802
liver	.674	.733	.739	.716	.724	.721	.739	.748
mw	.923	.910	.921	.918	.921	.921	.913	.921
pc1req	.731	.683	.750	.721	.731	.692	.654	.692
pc4	.900	.901	.908	.909	.899	.901	.908	.897
sonar	.813	.832	.832	.832	.803	.808	.827	.832
spect	.881	.885	.881	.885	.885	.885	.885	.881
spectf	.824	.813	.816	.820	.824	.816	.809	.794
ttt	.983	.991	.986	.991	.991	.991	.984	.982
wbc	.955	.961	.955	.961	.955	.957	.955	.952
vote	.867	.874	.870	.872	.874	.876	.870	.870
Mean	**.815**	**.816**	**.824**	**.822**	**.819**	**.814**	**.815**	**.816**
Rank	**4.81**	**4.52**	**3.42**	**3.46**	**4.29**	**4.94**	**5.21**	**5.35**

Here, it is seen that only three of the suggested combination methods produce higher mean accuracies than the random forest ensemble. Interestingly, it is in fact the most straightforward setups, i.e., *Prob*, *Rank* and *OOB-g* that all achieve higher accuracy, averaged over all data sets, than any of the underlying ensemble models obtain on their own. The three most elaborate schemes for combining ensemble predictions, i.e., *OOB-l*, *Gated* and *Tie-breaker*, on the other hand, all fail to improve overall accuracy. The highest mean accuracy is actually obtained by the simplest method (*Prob*), which performs better than both underlying ensembles on eight data sets, and worse than both of them on only a single data set. The mean ranks reflect the accuracy results, with the top three methods accuracy-wise having the lowest three means ranks.

In order to determine any statistically significant differences, we used the procedure recommended in [36] and performed a Friedman test [37], followed by Bergmann-Hommel's [38] dynamic procedure to establish all pairwise differences. Unfortunately, although the Friedman test shows that there are statistically significant differences, the n vs. n *post-hoc* test does not reveal any pairwise differences. One explanation is the fact that we evaluate as many as eight techniques on relatively few data sets. With this in mind, we also performed a 1 vs. n *post-hoc* test, treating *Prob* as the control. Here Hochberg's procedure [39] was used, and the resulting adjusted p-values are shown in Table III below. According to this test, *Prob* is indeed significantly more accurate (for $\alpha = 0.05$) than *Tie-breaker*, but no other differences are statistically significant.

TABLE III. ADJUSTED p-VALUES FOR ACCURACY RESULTS. *Prob* VS. OTHER TECHNIQUES USING HOCHBERG'S PROCEDURE.

Technique	p-value
Tie-br.	0.043
Gated	0.068
OOB-l	0.169
ANN	0.194
RF	0.355
OOB-g	0.410
Rank	0.953

Table IV below shows the ranking ability of the different techniques, given as AUC.

TABLE IV. AUC RESULTS

	ANN	RF	Prob	Rank	OOB-g	OOB-l	Gated	Tie-br.
colic	.879	.882	.881	.880	.882	.881	.877	.877
creditA	.923	.933	.933	.924	.932	.932	.931	.933
diabetes	.840	.829	.843	.836	.838	.835	.833	.836
german	.633	.625	.638	.632	.633	.640	.617	.638
haberman	.692	.674	.686	.701	.692	.684	.683	.686
heartC	.911	.905	.910	.914	.908	.908	.915	.895
heartH	.914	.896	.912	.903	.905	.900	.905	.912
heartS	.892	.894	.903	.904	.888	.891	.894	.903
hepati	.881	.900	.894	.904	.886	.883	.869	.887
iono	.974	.982	.984	.983	.982	.982	.978	.976
je4042	.786	.809	.815	.805	.809	.804	.812	.815
je4243	.676	.712	.723	.718	.712	.711	.705	.710
kc1	.684	.713	.720	.698	.698	.695	.707	.718
kc2	.839	.835	.848	.842	.834	.834	.842	.848
liver	.728	.754	.769	.744	.745	.740	.761	.764
mw	.797	.841	.838	.829	.802	.802	.786	.838
pc1req	.740	.743	.755	.750	.740	.708	.697	.730
pc4	.906	.947	.946	.930	.933	.937	.945	.946
sonar	.914	.957	.940	.919	.948	.945	.929	.915
spect	.699	.699	.703	.715	.699	.699	.715	.703
spectf	.852	.839	.852	.848	.852	.845	.838	.852
ttt	.999	1.000	.999	.996	1.000	1.000	.998	.999
wbc	.983	.980	.981	.984	.982	.983	.984	.978
vote	.924	.917	.927	.921	.914	.916	.918	.927
Mean	**.830**	**.839**	**.845**	**.839**	**.836**	**.834**	**.834**	**.840**
Rank	**5.63**	**4.44**	**2.60**	**4.08**	**4.56**	**5.25**	**5.38**	**4.06**

For AUC, the mean results differ slightly from the mean accuracies obtained. First of all, it should be noted that the ANN ensemble performs much worse on mean AUC than the random forest. Turning to the proposed combination methods, *Prob* once again performs best, and is the clear winner, both on mean AUC and on mean ranks. No other proposed combination technique outperforms random forests on mean AUC, although *Rank* and *Tie-breaker* achieve slightly lower mean ranks. For the AUC results, the differences are substantial enough for the n vs. n Bergmann-Hommel's *post-hoc* test to identify that *Prob* obtained significantly higher AUC than *ANN*, *OOB-l* and *Gated*. Using Hochberg's procedure for a 1 vs. n *post hoc* test, *Prob* is, in fact, seen to be significantly better than all other techniques, shown by the modified p-values in Table V below.

TABLE V. ADJUSTED p-VALUES FOR AUC RESULTS. *Prob* VS. OTHER TECHNIQUES USING HOCHBERG'S PROCEDURE.

Technique	p-value
ANN	1.4E-04
Gated	5.3E-04
OOB-l	9.1E-04
OOB-g	0.022
RF	0.029
Rank	0.039
Tie-br.	0.039

The overall result regarding predictive performance is thus that the most straightforward combination method, *Prob*, emerges as the clear winner, having both the highest mean accuracy and AUC and lowest mean rank of all setups. This implies that a simple combination procedure is available for increasing predictive performance in ensemble learning; build two highly accurate ensembles with different techniques and utilize their probability estimates to select the predicting ensemble for each new test instance. None of the more complicated setups perform consistently well, as seen by e.g. *Tie-breaker* having the second best results on AUC, but being worst on accuracy ranks.

In order to gain insights into how the suggested setups function in practice, it is useful to investigate how the two ensemble models are used to perform the actual classifications. Table VI below shows the proportion of instances predicted by the random forest, for each data set.

TABLE VI. PROPORTION OF THE PREDICTIONS MADE BY THE RANDOM FOREST

	Prob	Rank	OOB-g	OOB-l	Gated
colic	.213	.510	.697	.627	.437
creditA	.348	.504	.900	.803	.423
diabetes	.573	.525	.201	.310	.503
german	.591	.553	.000	.158	.467
haberman	.647	.523	.000	.131	.643
heartC	.301	.520	.199	.315	.325
heartH	.485	.560	.201	.259	.348
heartS	.333	.552	.400	.430	.281
hepati	.413	.594	.606	.542	.510
iono	.560	.563	1.000	.897	.546
je4042	.778	.522	1.000	.837	.881
je4243	.738	.474	1.000	.658	.865
kc1	.624	.542	.451	.442	.857
kc2	.672	.561	.100	.209	.797
liver	.748	.519	.900	.651	.924
mw	.660	.551	.100	.111	.478
pc1req	.731	.596	.000	.250	.663
pc4	.658	.519	.700	.640	.649
sonar	.135	.553	.803	.663	.438
spect	.560	.592	1.000	.986	.390
spectf	.356	.543	.000	.184	.363
ttt	.005	.529	1.000	.994	.054
wbc	.585	.553	.400	.400	.335
vote	.375	.497	.899	.805	.337
Mean	**.508**	**.540**	**.520**	**.510**	**.535**

The immediate impression from the mean values is that the two ensembles contribute more or less equally to the final prediction, albeit with a slight bias towards the random forest. Since this ensemble has slightly better predictive performance than the bagged ANNs, this is an encouraging indication that the basic idea works in practice. Closer inspection of the results does, however, reveal dramatic variations, both within each setup and each data set. Strikingly, for some data sets, e.g., *ttt*, the proportion of instances classified by the random forest can vary over the different setups from almost 0% to 100%. Similarly, for specific setups like *OOB-g*, the variation between data sets is also from zero to all instances. Again, this is an indication that the different setups successfully select the ensemble most suited to perform the prediction. Finally, it is noteworthy that *Rank* is the most stable setup, and also has the highest mean proportion of instances predicted by the random forest.

V. CONCLUDING REMARKS.

We have in this paper evaluated procedures combining two ensembles, one random forest and one ensemble of bagged neural networks, for predictive classification. The underlying assumption is that although both ensembles have similar and very high accuracy, they still disagree on a fairly large proportion of all test instances. With this in mind, it should be possible to increase the accuracy, compared to using either of the ensembles, by somehow combining them. In the experimentation, a number of straightforward approaches was evaluated, and the results show that it is indeed possible to devise combination strategies capable of outperforming the standard ensembles. The best combination strategy turned out to be one of the most straightforward, i.e., simply using the ensemble that is the most certain, for each test instance. This strategy obtained the best ranking for both accuracy and AUC, with the differences to all other methods being statistically significant for AUC.

REFERENCES

[1] T. G. Dietterich, "Ensemble methods in machine learning," in *Multiple Classifier Systems*, 2000, pp. 1–15.

[2] D. W. Opitz and R. Maclin, "Popular ensemble methods: An empirical study," *J. Artif. Intell. Res. (JAIR)*, vol. 11, pp. 169–198, 1999.

[3] T. G. Dietterich, "Machine-learning research: Four current directions," *The AI Magazine*, vol. 18, no. 4, pp. 97–136, 1998.

[4] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[5] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 8, pp. 832–844, 1998.

[6] G. Brown and L. I. Kuncheva, ""good" and "bad" diversity in majority vote ensembles," in *Multiple Classifier Systems*. Springer, 2010, pp. 124–133.

[7] A. Krogh and J. Vedelsby, "Neural network ensembles, cross validation, and active learning," *Advances in Neural Information Processing Systems*, vol. 2, pp. 231–238, 1995.

[8] G. Brown, J. Wyatt, R. Harris, and X. Yao, "Diversity creation methods: a survey and categorisation," *Information Fusion*, vol. 6, no. 1, pp. 5–20, 2005.

[9] L. I. Kuncheva, "That elusive diversity in classifier ensembles," in *Pattern Recognition and Image Analysis*. Springer, 2003, pp. 1126–1138.

[10] L. Saitta, "Hypothesis diversity in ensemble classification," in *Foundations of Intelligent Systems*. Springer, 2006, pp. 662–670.

[11] E. K. Tang, P. N. Suganthan, and X. Yao, "An analysis of diversity measures," *Machine Learning*, vol. 65, no. 1, pp. 247–271, 2006.

[12] U. Johansson, T. Lofstrom, and H. Bostrom, "Overproduce-and-select: The grim reality," in *Computational Intelligence and Ensemble Learning (CIEL), 2013 IEEE Symposium on*. IEEE, 2013, pp. 52–59.

[13] L. Breiman, "Bagging predictors," *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.

[14] U. Johansson and T. Lofstrom, "Producing implicit diversity in ann ensembles," in *Neural Networks (IJCNN), The 2012 International Joint Conference on*. IEEE, 2012, pp. 1–8.

[15] B. Efron, "Computers and the theory of statistics: thinking the unthinkable," *SIAM review*, vol. 21, no. 4, pp. 460–480, 1979.

[16] M. Stone, "An asymptotic equivalence of choice of model by cross-validation and akaike's criterion," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 44–47, 1977.

[17] K.-C. Li, "From stein's unbiased risk estimates to the method of generalized cross validation," *The Annals of Statistics*, pp. 1352–1377, 1985.

[18] D. H. Wolpert, "Stacked generalization," *Neural networks*, vol. 5, no. 2, pp. 241–259, 1992.

[19] S. Džeroski and B. Ženko, "Is combining classifiers with stacking better than selecting the best one?" *Machine learning*, vol. 54, no. 3, pp. 255–273, 2004.

[20] K. M. Ting and I. H. Witten, "Issues in stacked generalization," *J. Artif. Intell. Res.(JAIR)*, vol. 10, pp. 271–289, 1999.

[21] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Effective voting of heterogeneous classifiers," in *Machine Learning: ECML 2004*. Springer, 2004, pp. 465–476.

[22] G. Tsoumakas, L. Angelis, and I. Vlahavas, "Selective fusion of heterogeneous classifiers," *Intelligent Data Analysis*, vol. 9, no. 6, pp. 511–525, 2005.

[23] M. Woźniak, M. Graña, and E. Corchado, "A survey of multiple classifier systems as hybrid systems," *Information Fusion*, vol. 16, pp. 3 – 17, 2014.

[24] R. Caruana, A. Niculescu-Mizil, G. Crew, and A. Ksikes, "Ensemble selection from libraries of models," in *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004, p. 18.

[25] D. Ruta and B. Gabrys, "Classifier selection for majority voting," *Information fusion*, vol. 6, no. 1, pp. 63–81, 2005.

[26] G. Giacinto and F. Roli, "Design of effective neural network ensembles for image classification purposes," *Image and Vision Computing*, vol. 19, no. 9, pp. 699–707, 2001.

[27] A. S. Britto, R. Sabourin, and L. E. Oliveira, "Dynamic selection of classifiers—a comprehensive review," *Pattern Recognition*, vol. 47, no. 11, pp. 3665–3680, 2014.

[28] G. Giacinto and F. Roli, "Dynamic classifier selection based on multiple classifier behaviour," *Pattern Recognition*, vol. 34, no. 9, pp. 1879–1881, 2001.

[29] L. Didaci, G. Giacinto, F. Roli, and G. L. Marcialis, "A study on the performances of dynamic classifier selection based on local accuracy estimation," *Pattern Recognition*, vol. 38, no. 11, pp. 2188–2191, 2005.

[30] A. H. Ko, R. Sabourin, and A. S. Britto Jr, "From dynamic classifier selection to dynamic ensemble selection," *Pattern Recognition*, vol. 41, no. 5, pp. 1718–1731, 2008.

[31] E. M. Dos Santos, R. Sabourin, and P. Maupin, "A dynamic overproduce-and-choose strategy for the selection of classifier ensembles," *Pattern Recognition*, vol. 41, no. 10, pp. 2993–3009, 2008.

[32] K. Bache and M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[33] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, and S. García, "Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework," *Multiple-Valued Logic and Soft Computing*, vol. 17, no. 2-3, pp. 255–287, 2011.

[34] T. Fawcett, "An introduction to roc analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006.

[35] A. P. Bradley, "The use of the area under the roc curve in the evaluation of machine learning algorithms," *Pattern Recognition*, vol. 30, no. 7, pp. 1145–1159, 1997.

[36] S. Garcıa and F. Herrera, "An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons," *Journal of Machine Learning Research*, vol. 9, no. 2677-2694, p. 66, 2008.

[37] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *Journal of American Statistical Association*, vol. 32, pp. 675–701, 1937.

[38] B. Bergmann and G. Hommel, "Improvements of general multiple test procedures for redundant systems of hypotheses," in *Multiple Hypotheses Testing*. Springer, 1988, pp. 100–115.

[39] Y. Hochberg, "A Sharper Bonferroni Procedure for Multiple Tests of Significance," *Biometrika*, vol. 75, no. 4, pp. 800–802, 1988.

Named Entity Recognition in Affiliations of Biomedical Articles Using Statistics and HMM Classifiers

Jongwoo Kim and George R. Thoma

Abstract—**This paper proposes an automated algorithm that extracts authors' information from affiliations in biomedical journal articles in MEDLINE® citations. The algorithm collects words from an affiliation, estimates features of each word, and uses a supervised machine-learning algorithm called Hidden Markov Model (HMM) and heuristics rules to identify the words as one of seven labels such as city, state, country, etc. Eleven sets of word lists are collected to train and test the algorithm from 1,767 training data set. Each set contains collections of words ranging from 100 to 44,000. Experimental results of the proposed algorithms using a testing set of 1,022 affiliations show 94.23% and 93.44% accuracy.**

Keyword- Named Entity Recognition, HMM, Heuristic Rule, MEDLINE

I. Introduction

THE U.S. National Library of Medicine (NLM) maintains the MEDLINE database, a bibliographic database containing over 22 million citations related to the biomedical journal literature [1]. Each citation includes fifty-one different fields for each record. NLM receives journal article citations in XML format directly from journal publishers and adds additional fields to the record which are provided by journal article indexers. The number of citations in MEDLINE is rapidly increasing every year. NLM collects statistics such as the number of citations for each publication year, the number of citations of total year, the number of citations with authors in total year, etc. However, there are no detailed statistics such as, the number of citations published per each country each year, the number of citations per each organization each year, or the number of citations that received grants from NIH per each country each year, etc. In addition, there is no citation field for country, organization, etc. in the existing citations. Therefore, extraction of such fields from authors' affiliations is critical for the collection of detailed statistics.

There are several studies on extracting authors' information from texts. Robinson et al. extracted affiliations from free texts [2] and Kim et al. [3] extracted affiliations from journal articles. Further studies have been done to extract information from the authors' affiliations. Yu et al. [4] used word dictionaries and regular expression to extract three labels (institution, country and email address). Jonnalagadda et al. [5] used rules and word dictionaries to extract eight different labels. Torii et al. [6] used a HMM package to label eight different labels for words in affiliations. However, the papers did not address the following issues. First, labeling a word that contains two or three labels (no separators between labels). Second, labeling affiliations without country name. Third, labeling affiliations that contain more than two organization names. Fourth, usage of label orders and relationship between labels. Fifth, usage of probabilities of each word for each label.

Therefore, we propose a prototype of an automatic algorithm handling the above issues to classify words in affiliations into seven different labels to collect detailed statistics. HMM, statistics, and heuristic rules are adapted for the algorithms.

The remainder of this paper is organized as follows. Section II describes authors' affiliations in articles. The details of our methods are presented in Section III and IV. We discuss experimental results in Section V, and show conclusions in Section VI.

II. Authors' affiliations

Words in authors' affiliations can be categorized by two groups, organization and geographic terms. Organization terms include department, school, university, institute, etc. Geographic terms include city, state/province, postal code, and country. Email can belong to either one or two groups since it can contain country and organization names. There are several types of affiliations based on the orders of the term words. Some affiliations show just organization names and others show the full address (mailing address) of their organizations. Our final goal is to label affiliation words into nine different labels (Department, School, University, City, State/Province, Postal Code, Country, Email, and Other). In this paper, as a preliminary work, we label affiliation words into seven labels (University, City, State/Province, Postal Code, Country, Email, and Other). In private organizations, "University" means company names, "School" means institutes or centers that belongs to the companies, and "Department" means departments or divisions that belongs to the institutes or center.

Table I shows some examples of affiliations. In the table, PMID (PubMed Unique Identifier) is a unique reference number for the MEDLINE citations. "Un" means "University", "Ci" means "City", "St" means "State", "Co" means "Country", "Po" means "Postal Code", "Em" means "Email", and "Ot" means "Other". Some affiliations have a company name only (Type 1), some have full address including street name (Type 2), some have a country name

J. Kim is with the National Library of Medicine, Bethesda, MD 20893, USA (corresponding author to provide phone: 301-435-3227; fax 1 301 402-0341; e-mail: jongkim@mail.nih.gov).

G. R. Thoma is with the National Library of Medicine, Bethesda, MD 20893, USA

(Type 3), some do not have a country name (Type 4), and others have an email address (Type 5). We define the type based on the label orders and use it to develop the algorithm.

TABLE I
AFFILIATIONS IN ARTICLES IN MEDLINE.

Type	Explanation/Examples	Label Order	PMID
1	bioMerieux, Inc.	Un	23002511
2	Faculty of Kinesiology, University of Calgary, 2500 University Drive NW, Calgary, Alberta, Canada T2N 1N4.	Ot, Un, Ci, St, Co, Po	23000101
3	Department of Urology, School of Medicine, University of Gaziantep, 27310 Gaziantep, Turkey.	Ot, Ot, Un, Po, Ci, Co	23001641
4	Department of Psychology, University of Houston.	Ot, Un	23000106
5	Department of Physics, The Ohio State University, Columbus Ohio 43210, USA.	Ot, Un, Ci, St, Po, Co	23002754
6	Department of Psychology, School of Life and Medical Sciences, University of Hertfordshire, UK. k.laws@herts.ac.uk	Ot, Ot, Un, Co,Em	23001963
7	Zaklad Medycyny Nuklearnej Pomorskiego Uniwersytetu Medycznego w Szczecinie ul. Unii Lubelskiej 1, 71-252 Szczecin.	Ot, Un, Po, Ci	23002662
8	Division of Gastroenterology, Hepatology, and Nutrition, University of Pittsburgh Medical Center, Pittsburgh, Pennsylvania, USA.	Ot, Un, Ci, St,, Co	23000233
9	Department of Global Safety Pharmacology, Department of Pharmacokinetics, Dynamics & Metabolism, and Neuroscience, Pfizer Global Research and Development Eastern Point Road, Groton, CT 06340, USA. anthony.fossa@icardiac.com	Ot, Un, Ci, Po, Co, Em	23000177

III. ALGORITHM WORKFLOW

The workflow of the proposed algorithm is as follows. First, replace words (names) with standardized words using a dictionary. Second, divide an affiliation into words using several separators. Third, divide a word with more than two labels into several words using geographic information (city, state/province, country, postal code, etc.) in collected word lists. Fourth, assign a possibility value for each word for each label using collected word lists and statistics. Fifth, classify each word as one of the seven labels using HMMs. More detailed information of each step is shown in the next section.

IV. PROPOSED APPROACH

We use the following word lists and algorithms for labeling words in affiliations.

A. Word Normalization

There are several abbreviated (or non-standard) words in affiliations. We first standardize the words. Table II shows some examples collected from a training set of 1,767 affiliations in MEDLINE. For example, authors use several ways to write the country name "China" as shown in the second row. All non-standard words are replaced with our own standard words and abbreviated names are replaced with full names also. For example, when the organization name "NIH" is replaced with "National Institutes of Health", it becomes clear that "NIH" is an institution name. 155 words are collected related to city, country, organization name, and other words for the list.

TABLE II
LIST OF WORDS FOR STANDARDIZATION

Standard Word	Non-Standard Word
China	P R China, People's Republic of China, PR of China, etc.
Germany	Deutschland, Federal Republic of Germany, F.R.G, etc.
Korea	Republic of Korea, South Korea
National Institutes of Health	NIH
Technische Universität Berlin	TU Berlin

B. Postal Code Detection

Every country has their postal codes formats. We search the codes of several countries using Google search engine [7], collect the codes for 121 countries, and save them as Regular Expression [8] formats. Table III shows the formats of some countries.

TABLE III
LIST OF POSTAL CODE FORMATS OF COUNTRIES

Country Name	Postal Code (Regular Expression)
Austria	\\b((A-|)[0-9O]{4})\\b
Brazil	\\b(([0-9O]{5}|[0-9O]{2}[.][0-9O]{3})[-][0-9|O]{3})\\b
India	\\b(([0-9O]{6})|([0-9O]{3}[ ][0-9O]{3}))\\b
USA	\\b([A-Z]{2}(|)[0-9O]{5}[-][0-9O]{4})\\b
Vietnam	\\b([0-9O]{6})\\b

C. City, Region, and Country names Detection

A list of city, state, and country names are necessary to recognize them from affiliations. First, we collect names from affiliations in MEDLINE. In addition, we use Google search engine [7] to collect more information. There are about 44,000 names in the list. Table IV shows examples of some of the data.

TABLE IV
LIST OF CITY, REGION, AND COUNTRY NAMES

Country	Region (State/Province)	City/Town
Australia	New South Wales	Sydney
Canada	Quebec	Montreal
China	Shaanxi	Xi'an
Germany	Nordrhein-Westfalen	Düsseldorf
Germany	North Rhine-Westphalia	Dusseldorf
Philippines	Sorsogon	San Juan
South Africa	Eastern Cape	Grahamstown
Spain	Pontevedra	Vigo
SAUDI Arabia	Makkah	Thuwal
USA	Maryland	Bethesda

D. Organization Name Words

Organization names are categorized into three labels (Department, School, and University levels). It is clear when affiliations are from universities. However, it is hard to categorize affiliations from companies, laboratories, etc. Therefore, we search for affiliations in the training set, classify them into the three labels using the Google search engine, and collect the words related to the three labels as shown in Table V. Among them, several words are used in more than two labels. Table VI shows the probabilities of the three labels for some words in Table V. For example, "Center" is used for University (University level) 51 times, School 31 times, and Department 15 times. The probabilities are used to classify organization names at the University level.

TABLE V
LIST OF WORDS FOR ORGANIZATION NAMES

Academy	Fachbereich	Laboratorios
Aquarium	Katedra	Laboratorie
Agence	Division	Laboratoria
Agency	Engineering	Laboratorium
Association	Faculty	Laboratory
Branch	Faculte	Laboratório
Bureau	Faculté	Laboratorio
Campus	Fakultät	Laboratoire
Center	Faculdade	Library
Centre	Facultad	Limited
Centro	Faultad	LLC
Központ	Facoltà	Ministry
Centrum	Kar	Museum
BioCenter	UFR	Organization
Hemocentro	Wydział	Organisation
Herzzentrum	Wydzial	Pharmaceutical
Clinic	Foundation	Pharma
Clinics	Fundación	Pharmacal
Clinico	Fund	Pharmaceutica
Clínico	Group	Pty
Clínica	Hospice	School
Klinik	Hospices	Services
Klinika	Hospital	Society
Kliniki	Hospitals	Trust
Poliklinigi	Hôpital	University
College	Hôpitaux	Universitat
Collegium	Hospitalier	Universiti
Hochschule	Klinikum	Université,
Charities	Ziekenhuis	Universite
Company	Ospedale	Università
Commission	Ospedaliera	Universitaria
Committee	Ospedaliero	Universität
Corporation	Hastanesi	Universiteit
Council	Incorporated	Universidad
Consiglio	INC	Universidade
Department	Inc	Universitaire
Département	Inc.	Universitätsspital
Departments	Institutes	Universitätsklinikum
Departament	Institution	Universitätskliniken
Département	Institute	Universitätsmedizin
Departement	Institut	Uniwersytetu
Dipartimento	Instituto	Uniwersytet
Departamento	Instytut	Nationale Supérieure
Deparment	Institutet	Egyetem
Dept.	Istituto	Tudományegyetem
Dept	Intézet	Unit
Dpto	Laboratories	Unité

TABLE VI
PROBABILITIES OF WORDS FOR UNIVERSITY, SCHOOL, AND DEPARTMENT LABEL

Affiliation Words	Prob. of University	Prob. of School	Prob. of Department
Center, Centre, Centro, Központ, etc.	0.5258	0.3196	0.1546
Department, Département, Département, Dipartimento, etc.	0.0043	0.0239	0.9717
Faculty, Faculte, Faultad, Facoltà, etc.	0.0625	0.8906	0.0469
Hospital, Hôpital, Hôpitaux, Klinikum, Hastanesi, etc.	0.7383	0.2523	0.0093
Institute, Institution, Institut, Intézet, etc.	0.4779	0.4412	0.081
Laboratory, Laboratorios, Laboratorium, Laboratoire, etc.	0.0833	0.3000	0.6167
University, Universitat, Universitaria, Uniwersytet, etc.	0.9795	0.0154	0.0051

E. Other Words

There are words such as road name, building number, subdivision name, etc. in affiliations. These words are labeled as Other. Table VII shows some words collected for the Other label.

TABLE VII
LIST OF WORDS FOR OTHER LABEL

Other Word Category	Words
Road	Avenue, Avenida, Freeway, Route, Street,
Sub division	Ro, Ku, Gu, etc.
Building	Suit, Building,
P.O. Box	P.O.Box, PO Box, POB, Private Bag, etc.

F. Email Address

There are several Regular expression formats to recognize email addresses in affiliations. Among them, we use the following format of Regular Expression [9].

```
"([a-zA-Z0-9_\\-\\.]+)@((\\[[0-9]{1,3}\\.[0-9]{1,3}\\.[0-9]{1,3}\\.)|(([a-zA-Z0-9\\-]+\\.)+))([a-zA-Z]{2,4}|[0-9]{1,3})(\\]?)"
```

G. Abbreviate Organization Name Detection and Removal

Some authors write their organization names twice in affiliations; including full and abbreviated names. Table VIII shows some examples. Abbreviated names are usually enclosed in parentheses as shown in Type 1 to 5. However, all of them are not duplicated names. Type 6 shows that state name VA (province of Varese) is enclosed in parentheses. The following two methods are used to remove the abbreviated names before classifying words in affiliations.

TABLE VIII
AFFILIATIONS HAVING TWO SAME ORGANIZATION NAMES

Type	Affiliation with full and abbreviated organization names
1	Burn Research Center (BRC), Shahid Motahari Burns Hospital, Tehran University of Medical Science, Tehran, Iran. bsobooti@tums.ac.ir
2	Instituto de Biología Molecular y Celular (IBMC), Miguel Hernández University, 03202, Elche, Spain.
3	National Institute of Advanced Industrial Science and Technology (AIST), Umezono, Tsukuba 305-8568, Japan.
4	Laboratoire d'ergonomie et d'épidemiologie en santé au travail (LEEST), LUNAM Université, Université d'Angers, LEEST-UA InVS, Angers, France. celine.serazin@univ-angers.fr
5	Center of Calcium and Bone Research (COCAB), Mahidol University, Bangkok, Thailand.
6	U.O. Cardiologia-Emodinamica Istituto Clinico Humanitas Mater Domini, Via Gerenzano 2, Castellanza (VA), Italy. alielasi@hotmail.com

Method 1 works for the Types 1, 2, 3, and 4 and Method 2 works for Type 5. However, no method is working for the Type 6 case.

Method 1: (see Type 1 in Table VIII)

Step 1, replace several words (e.g., "d'é" to "E", "d'É" to "E", etc.) in an affiliation.

Step 2, find a name ("BRC") in a parenthesis or with all uppercase characters.

Step 3, collect all words using space, comma, semi-colon, colon, and parenthesis as separators.

Step 4, make a string ("BRCBSMBHTUMSTI") using the first uppercase character in each word.

skip a word if the first character in the word is not uppercase character.

Step 5, remove the name ("BRC") from the affiliation if it is found in the string in Step 4.

Method 2: (see Type 5 in Table VIII)

Step 1, replace several words (e.g., "d'é" to "E", "d'É" to "E", etc.) in an affiliation.

Step 2, find a name ("COCAB") in a parenthesis or with all uppercase characters.

Step 3, collect words using comma, semi-colon, colon, and parenthesis as separators.

For each word (having more characters than the name.)

Step 4, search each character in the name (COCAB) in the word ("Center of Calcium and Bone Research").

Step 5, remove the name from the affiliation if all characters in the name are found in the same order in the word and 75% of matched characters in the word are uppercase.

H. Word Separation

Seven separators (",", ";", ":", "(", ")", "[", "]") are used to separate words in an affiliation. The separators works well for most affiliations. However, many authors use "white space" as a separator frequently. Type 5 in Table I uses a "white space" separator between city, state, and postal code "Columbus Ohio 43210". Type 7 in Table I also uses a "white space" separator between department, school, and university. This causes labeling errors or increases computation time to separate the words in affiliations. In the case of Type 7 in Table I "Zaklad Medycyny Nuklearnej Pomorskiego Uniwersytetu Medycznego w Szczecinie ul. Unii Lubelskiej 1, 71-252 Szczecin", since there is no separator between department and university as is written in the Polish language, it causes word separation and University (organization name) labeling errors. This also creates errors in the Postal Code and City labels. Therefore, the following steps are used to separate words for accurate labeling.

Step 1. Divide words (w_i, where i=1 to n) in an affiliation using the seven separators.

Step 2. Search email.
  If w_i contains an email,
    Divide a word w_i into w_{ij}, where j=1 to k,
      using "white space".
    Replace w_i with w_{ij}, where j=1 to k.
    Update n=n+k-1.
    Stop.
  End If

Step 3. Search country name using Table IV.
  If w_{n-1} or w_n is not country name.
    For i=n to n-1
      Divide a word w_i into w_{ij}, where j=1 to m,
        using "white space" if no word found in Table V.
      If country name found in w_{ij},
        Replace w_i with w_{ij}, where j=1 to m.
        Update n=n+m-1.
        Stop.
      End If.
    End For
  End If.

Step 4. Search city and region names and postal code using Tables II, III and IV.
  If w_i is not city, region, or postal code.
    For i=n to 1
      Divide a word w_i into w_{ij}, where j=1 to p
        using "white space" if no word found in Table V.
      If one of the labels found in w_{ij},
        Replace w_i with w_{ij}, where j=1 to p.
        Update n=n+p-1.
        Stop.
      End If.
    End For
  End If

I. Adjust Possibilities of Labels

Heuristic rules are used to adjust possibilities of labels. All of the rules and ratios are based on the statistics obtained

from the training set. Table IX shows some of the rules. Rule 1 means if a word (w_i) does not have any clue for City, but the previous word (w_{i-1}) has a possibility for University and the next word (w_{i+1}) has a possibility for State, the word (w_i) has 98% of possibility for City.

TABLE IX
HEURISTIC RULES FOR ADJUSTING POSSIBILITY OF A LABEL

Rule	Condition
1	*If* $P_{University}(w_{i-1}) > 0$, $P_{City}(w_i) = 0$, and $P_{State}(w_{i+1}) > 0$, $Pcity(w_i)$=0.98.
2	If $P_{University}(w_{i-1}) > 0$, $P_{City}(w_i) = 0$, and $P_{Postal\ Code}(w_{i+1}) > 0$, $Pcity(w_i)$=0.80
3	If $P_{University}(w_{i-1}) > 0$ and $P_{University}(w_i) > 0$, $P_{University}(w_i) = P_{University}(w_i)\times 0.9384$. $P_{University}(w_{i-1}) = P_{University}(w_{i-1})\times 0.0616$.

J. Hidden Markov Model (HMM)

HMM [10] is used to extract labels from affiliations since it provides stable results in named entity recognition areas. In addition, the Viterbi [11] algorithm is used to finalize labels of words in affiliations from the HMM results. To train HMMs using the training set, we group the training set data by the label order and train HMMs for each group. The following is the complete algorithm procedure.

Step 1. Separate words from an input affiliation.
Step 2. Standardize words using Table II.
Step 3. Assign possibilities of Department, School, and University labels using Tables V and VI.
Step 4. Assign possibilities of City, State, Country, Postal Code, and Email labels using Tables III and IV. Assign 1.00 if a word is in the tables or meets formatting requirements.
Step 5. Assign possibilities of Other label using Table VII.
Step 6. Adjust possibility of labels using in Table IX.
Step 7. Apply all trained HMMs for the input affiliation and select one HMM (HMM_{final}) that has the highest value.
Step 8. Use the Viterbi algorithm in HMM_{final} to finalize labels of words in the affiliation.

V. EXPERIMENTAL RESULTS

All affiliations in MEDLINE in the ranges from PMID=23,000,000 to 23,004,000 are used in this experiment. Among them, 1,767 PMIDs (from PMID=23,000,000 to 23,002,000) are collected for the training set and 1,022 PMIDs (from 23,002,001 and 23,004.000) for the testing set. Since several PMIDs do not have any information in MEDLINE, training and testing sets do not have a similar amount of data.

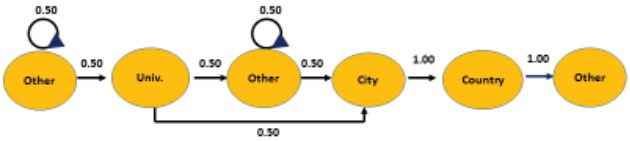
To train HMMs, we group the training set data by order of labels first and train HMMs for each group. To optimize the number of training HMMs and number of training data for each HMM, we remove "Other" between labels in the training set data. For example, a training set with data containing "Other, University, Other, City, Other, Postal Code, Other, Country" is assigned to "Other, University, City, Postal Code, Country" group for training.

We have 30 HMMs from the training set. Some HMMs contains a reasonable number of training data. However, other HMMs have less training data. Fifteen HMMs have less than ten training data. Table X shows some of the trained HMMs. The third row in the table shows that the HMM (Other, University, City, Country) has 280 training data and Fig. 1 shows the diagrams of the HMM. Fig. 1(a) shows the HMM from the training data (HMM Trained) and Fig. 1(b) shows the HMM modified from the HMM 1(a) (HMM Modified). i.e., the transition workflows from one label to other labels are the same in the two HMMs. The difference is that Fig. 1(b) has equal transition weights (=1/k) when one label can move to k different labels.

TABLE X
HMMS TRAINED USING THE TRAINING SET

HMM	**Number of PMIDs used**
Other,University,City,ZipCode,Country	311
Other,University,City,Country	280
Other,University,City,ZipCode,Country,Email	226
Other,University,City,Country,Email	210
Other,University,ZipCode,City,Country,Email	139
Other,University,City,State,Country	109
Other,University,City,State,ZipCode,Country	66
Other,University,Country,Email	37
Other,University,ZipCode,State,Country,Email	1

(a)



(b)

Fig. 1. (a) HMM (HMM Trained) in the third row in Table IX. (b) HMM (HMM Modified) modified from the HMM (a).

Table XI shows the test results. We consider errors when one of the words in an affiliation is miss-labeled by HMMs. The HMMs Trained column shows 94.23% accuracy and HMMs Modified shows 93.35% accuracy. The HMMs Trained shows better performance than the HMMs Modified.

We evaluate the 58 errors from HMMs Trained and categorize them into five as shown in the Table XII. First, the major errors are caused by a word separator problem. As shown in the second row in the table, there are no separators between department and university, and between postal code and city

name. It is written in Polish and there is no country name there. In addition, our word lists (Tables V and VI) do not have much information about non-English languages. All these issues make the algorithm more difficult to separate the words. Second, there are not many organization names in our word lists. "IFW Dresden" is the organization (third row). However, the name does not contain any clue word related to organization names. Therefore, "Institute for Integrative Nanosciences" is recognized as the organization name. Third, there is no trained HMM that fits to the input (fourth row). This problem can be resolved by increasing the size of training set data. Fourth, existing HMMs are trained for processing single affiliation. Therefore, the algorithm cannot handle texts with multiple affiliations (fifth row). The last error is the order of labels (sixth row). 93.84% of affiliations in the training set has the order of "Department, School, University". However, University word "USDA Forest Service" comes first and Department word "Aldo Leopold Wilderness, Research Institute" comes later. Since there is no clear word representing University in "USDA Forest Service", but the word "Institute" is found in "Aldo Leopold Wilderness, Research Institute", "Aldo Leopold Wilderness, Research Institute" is labeled as University label. This problem can be resolved by collecting names that belong to University label and use them for labeling.

TABLE XI
PERFORMANCE OF THE PROPOSED HMMS

HMM Mode	HMMs Trained	HMMs Modified
Total	1,022	1,022
True	964	955
False	58	67
Accuracy	94.32%	93.44%

TABLE XII
ERROR ANALYSIS

Error Analysis	Number of PMIDs	Affiliation Example
Word separation error	21	Zaklad Medycyny Nuklearnej Pomorskiego Uniwersytetu Medycznego w Szczecinie ul. Unii Lubelskiej 1, 71-252 Szczecin.
Hard to recognize words in University label	20	Institute for Integrative Nanosciences, IFW Dresden, D-01069 Dresden, Germany. j.zhang@ifw-dresden.de
HMM does not exit	11	Department of Physics, Indian Institute of Technology, Bombay, Powai, Mumbai-400 076, India. supravat@phy.iitb.ac.in
Multiple affiliations	5	Heart Institute, Ha'Emek Hospital, Afula, Israel, affiliated with Rappaport Faculty of Medicine, Haifa, Israel.
Label order problem	1	USDA Forest Service, Rocky Mountain Research Station, Aldo Leopold Wilderness, Research Institute, 790 East Beckwith, Missoula, MT 59801, USA. sean_parks@fs.fed.us

VI. CONCLUSIONS

This paper proposes an automatic algorithm to classify seven different labels from affiliations in biomedical journal articles using statistics, heuristic rules and HMM. We collect seven different word list tables to estimate the possibilities of seven different labels for each word in the author affiliations. We also collect 1,767 affiliations for a training set and 1,022 affiliations for a testing set from MEDLINE.

The proposed module performs relatively well. The results shows 94.23% accuracy from HMMs Trained and 93.35% accuracy from HMMs Modified.

As a future task, we plan to use more data for the training set to handle additional different types of affiliations and collect (international) organization names for more accurate classification. In addition, we will extend the algorithm classifying the nine different labels from affiliations.

ACKNOWLEDGMENT

This research was supported by the Intramural Research Program of the National Institutes of Health, National Library of Medicine, and Lister Hill National Center for Biomedical Communications.

REFERENCES

[1] http://www.nlm.nih.gov/pubs/factsheets/medline.html.
[2] Chinchor, N, Robinson, P, "MUC-7 named entity task definition", Proceedings of the 7th Message Understanding Conference, 1997.
[3] Kim J, Le DX, Thoma GR. "Automated Labeling Of Biomedical Online Journal Articles", SCI 2005. Proc 9th World Multiconference on Systemics, Cybernetics and Informatics; 2005 Vol. 4; Orlando (FL).
[4] Yu, W., Yesupriya, A. et. al., "An Automatic method to generate domain-specific investigator networks using PubMed abstracts', BMC Medical Informatics and Decision Making, Vol 7, 17, 207.
[5] Jonnalagadda, S. and Topham, p., "NEMO: Extraction and normalization of organization names from PubMed affiliation strings", Journal of Biomedical Discovery and Collaboration, Vol. 5, 50-75, 2010.
[6] Torii, M.,Wagholikar, K., Kim, D., Liu, H. "Named Entity Recognition in the MEDLINE Affiliation Field: A Step towards Enhanced Maintenance of Researcher Profile Systems" AMIA CRI, pp. 164, 2012.
[7] http://www.google.com.
[8] https://msdn.microsoft.com/en-us/library/hs600312(v=vs.110).aspx.
[9] http://regexlib.com.
[10] Chahramani, Z., "An Introduction to Hidden Markov Models and Bayesian Networks", International Journal of Pattern Recovnition and Artificial Intelligence, 15 (1), pp. 9-42, 2001.
[11] Viterbi, A. J., Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," *IEEE Transactions on Information Theory* , vol. IT-13, April, 1967, pp. 260-269.

ISBN: 1-60132-431-6, CSREA Press ©

SESSION

LATE POSTERS

Chair(s)

TBA

ISBN: 1-60132-431-6, CSREA Press ©

ISBN: 1-60132-431-6, CSREA Press ©

Real-Time Classification of User Clicked News for Category Proportion Adjustable News Recommendation

Zhegao Piao, Seong Joon Yoo and Yeong Hyeon Gu

Abstract—**There have been studies on personalized news recommendation using collaborative filtering mechanism based on users' click behaviors. However, few existing studies have focused recommending news depending on the rates of the news categories interests. In this paper, we present a personalized news recommendation system which builds profiles of users' news categories interests, determines the number of news articles to be recommended for each news category in proportion to news categories interests and ranks news articles according to the user's news interests. In order to find the news categories that are interesting to read, the smart device collects the web pages viewed by the user and classifies the contents. We use machine learning techniques in order to classify web pages into different categories and experimental results show that Naïve Bayes achieved the highest F-measure. The news preference is automatically calculated by the news reading time and the length of the news text. News articles with high preference generated by the collaborative filtering technique are recommended based on the rate of each category. We implemented recommendation system based on collaborative filtering using Mahout Recommender API on Hadoop. Through user testing, we also assess whether the proposed system is a useful.**

Proposed Methods and Results

As smart devices became widely available, the focus of web services has moved from the PC to smart devices. It is the same for news services. Moreover, with the development of news recommendation systems, the demand is growing among users to be recommended with news items in the preferred categories. Moreover, users want the amount of news items recommended to be automatically adjusted according to the extent of preference.

The existing mobile news recommendation systems cannot automatically adjust the amount of news. Furthermore, no technology that automatically adjusts the amount of news recommended according to the extent of preference is available in the academic papers[1]-[9] or mentioned in various news recommendation systems[10]-[15] that deal with news yet. This paper proposes methods by which to sort news items in each field according to the rates of the categories preferred by the user, to adjust the amount of news, and to calculate the preference value of the news recommended by the user's news consumption pattern. Using machine learning, the categories preferred by the user and the rates of the applicable categories are calculated after classifying the web pages recently viewed by the user. New news items in the applicable categories are recommended according to these preferences and the related rates. For example, if the user viewed news items in the political field 50% of the time, in the Entertainment field 30% of the time and in other categories 20% of the time during the past two to three days, the news items in the applicable categories are shown in the rates of 50%, 30% and 20%, respectively. In the event the user's news click rates changed to 30% for Politics, 50% for Entertainment and 20% for Miscellaneous, our system recommends news items in each field according to these rates.

One of the existing methods to recommend news to users of smart devices is to mix news items from different categories. However, with this method, the user must select those news items if the user wants to read those news items in any one specific field such as Sports, which is inconvenient. Another method is that news items are sorted and recommended by each field, but the editor determines the allocation of each recommended field, the types of news items and the volume. If this method is used, there could be instances in which the user may not be able to view those news items in the specific field such as Sports even though he or she want to read more news items in the Sports field.

Additionally, in [3], in order to recommend news items, a proposal was made to predict news items that the user may be interested in by using the Bayesian framework. In our paper, news items are recommended using the collaborative filtering method. With the existing collaborative filtering method, news items that other people like are recommended unilaterally. If this happens, the volume of news items that are recommended according to the user's preferences is not derived dynamically. However, with the method proposed in this paper, news items that are recommended by collaborative filtering are again filtered on the basis of the user's preferences.

The method used in this paper is shown in Fig.1. The method used in this paper is shown in Fig.1. First, step ① illustrates the user's click-behavior information is collected from the user's smart device. Then, the text contents of the web pages that the viewer has visited in the past are crawled. Then the collected contents are classified into eight categories

This research was supported by the MSIP(Ministry of Science, ICT and Future Planning), Korea, under the Global IT Talent support program (IITP-2016-H0905-15-1005) supervised by the IITP(Institute for Information and Communication Technology Promotion).

Z. Piao is with the Department of Computer Engineering, Sejong University, Seoul, Korea (e-mail: piaozhegao5@gmail.com).

S. J. Yoo is with the Department of Computer Engineering, Sejong University, Seoul, Korea (corresponding author; phone: +82-2-3408-3755; e-mail: sjyoo@sejong.ac.kr).

Y. H. Gu is with the Department of Computer Engineering, Sejong University, Seoul, Korea (e-mail: yhgu@sejong.ac.kr).

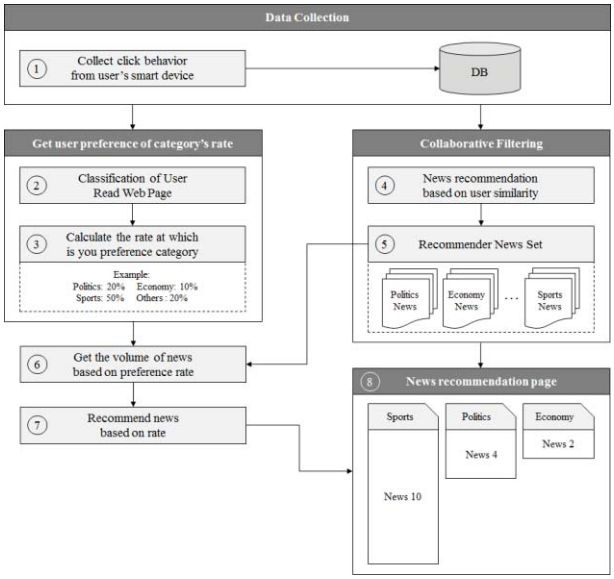


Fig. 1. Flowchart of the Proposed Personalized News Recommender

using machine learning, as shown in step ②. The classification results are then tabulated in order to find the categories preferred by the user, and the rate of each field is calculated, as shown in step ③. With the click-behavior information collected from the user's smart device, with collaborative filtering we recommend the news items preferred by the user based on the news items viewed by similar users using the Mahout Recommender API in Hadoop. This is shown in step ④. Step ⑤ shows the recommendation results are sorted by field and then according to the news preference value in each field to generate the recommended news data set. In order to provide better recommendations, preference values of the news items read by the user are required. However, users do not like to enter preference values after they finish reading news items. Therefore, a module to automatically enter preference values is required. We resolved this issue with a method whereby the time the user takes to read news items, the length of sentences with which to automatically evaluate news items and enter preference values. Lastly, the volume of news items to be shown in each field is calculated from the news items as shown in step ⑥ in the news set in step ⑤ and the rates of the categories preferred by the user. ⑦ shows news items are recommended according to the rates of the categories preferred by the user in order to show first the categories preferred by the user in the same web page and to show more recommended news items in the field.

With collaborative filtering, the cold-start issue occurs when the system is being used for the first time since there is no user log data. With the existing news sites[10], [11], recommendations start after the user directly clicks and selects his or her preferred categories. However, with this method the user must enter the preferred categories directly, so this method could be inconvenient for the use. In some news sites, news items are shown randomly. With the application proposed by us, when the application is initially installed without initial recommendations, the click behavior is collected from the user's smart device, and this information is sent to the server. From the click-behavior information received from the server, the categories preferred by the user and the rates are calculated. The cold-start problem was resolved by showing news items based on these rates.

Based on the result of classifying the contents of web pages viewed by the user in the past using six machine learning algorithms, Naïve Bayes, K-Nearest Neighbor, AdaBoost, Decision Tree, Artificial Neural Network, Support Vector Machine, the F-measure value of Naïve Bayes was the highest at 86.8%. Therefore, we classified the user's past data using Naïve Bayes. Additionally, we conducted a user-satisfaction test regarding the method of showing recommendation results according to the user's preference rates, thereby proving that the system we developed is useful.

REFERENCES

[1] H. Zhu, E. Chen, H. Xiong, K. Yu, H. Cao, and J. Tian, "Mining mobile user preferences for personalized context-aware recommendation," ACM Transactions on Intelligent Systems and Technology (TIST), vol.5, no.4, 2014.
[2] T. Ma, L. Guo, M. Tang, Y. Tang, M. Al-Rodhaan, and A. Al-Dhelaan, "A Collaborative filtering recommendation algorithm based on hierarchical structure and time awareness," IEICE Transactions, 2016
[3] J. Liu, P. Dolan, and E.R. Pedersen, "Personalized news recommendation based on click behavior," In Proc. of the 15th International Conference on Intelligent User Interfaces, pp.31-40, 2010.
[4] K. Choi and Y. Suh, "A new similarity function for selecting neighbors for each target item in collaborative filtering," Knowledge-Based Systems, vol.37, pp.146-153, 2013.
[5] B. Sarwar, K. George, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," In Proc. of the ACMInternational Conference on World Wide Web, pp. 285-295, 2001.
[6] J. Bobadilla, F. Ortega, A. Hernando, and J. Bernal, "A collaborative filtering approach to mitigate the new user cold start problem," Knowledge-Based Systems, vol.26, pp.225-238, 2012.
[7] B. Lika, K. Kolomvatsos, and S. Hadjiefthymiades, "Facing the cold start problem in recommender systems," Expert Systems with Applications, vol. 41, pp. 2065-2073, 2014.
[8] J. Bobadilla, F. Ortega, A. Hernanado, and A. Gutiérrez, "Recommender systems survey," Knowledge-Based Systems, vol.46, pp.109-132, 2013.
[9] Y.H. Gu, S.J. Yoo, Z. Piao, H. Yinhelin, Z. Jiang, and J.H. Park, "User preference analysis and visualization through the browsing history of smart devices", 2015.
[10] Ziny news (2016. 03. 05), http://www.zinynews.com/
[11] BBC (2016. 04. 07), http://www.bbc.com/news
[12] DAUM (2016. 04. 05) http://media.daum.net/
[13] BAIDU (2016. 03. 07), http://jian.news.baidu.com/
[14] GOOGLE (2016. 04. 02), https://news.google.co.kr/
[15] TIMES (2016. 04. 02), www.Times.com