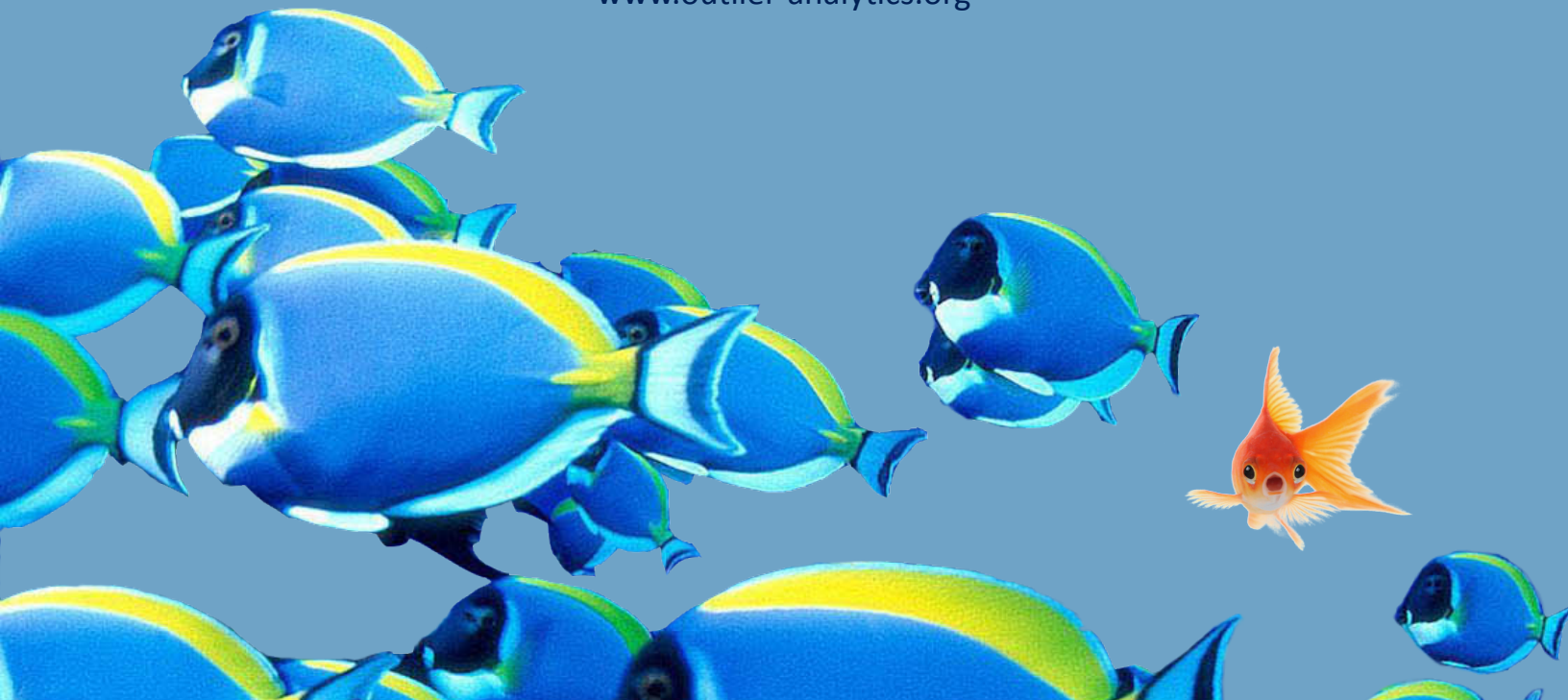
Proceedings of the
ACM SIGKDD Workshop on
Outlier Detection and Description

Leman Akoglu & Emmanuel Müller & Jilles Vreeken, editors

ODD 2013

Aug 11, Chicago, IL, USA

www.outlier-analytics.org



Proceedings of the
ACM SIGKDD Workshop on
Outlier Detection and Description

These proceedings are also included in the ACM Digital Library.

ODD'13, August 11, 2013, Chicago, IL, USA

ACM SIGKDD Workshop on Outlier Detection and Description

General Chairs

Leman Akoglu (Stony Brook University)
Emmanuel Müller (Karlsruhe Institute of Technology)
Jilles Vreeken (Universiteit Antwerpen)

Program Committee

Fabrizio Angiulli (University of Calabria)
Ira Assent (Aarhus University)
James Bailey (University of Melbourne)
Arindam Banerjee (University of Minnesota)
Albert Bifet (Yahoo! Labs Barcelona)
Christian Böhm (LMU Munich)
Rajmonda Caceres (MIT)
Varun Chandola (Oak Ridge National Laboratory)
Polo Chau (Georgia Tech)
Sanjay Chawla (University of Sydney)
Tijl De Bie (University of Bristol)
Christos Faloutsos (Carnegie Mellon University)
Jing Gao (University of Buffalo)
Manish Gupta (Microsoft (India)
Jaakko Holmén (Aalto University)
Eamonn Keogh (University of California – Riverside)
Matthijs van Leeuwen (KU Leuven)
Daniel B. Neill (Carnegie Mellon University)
Naren Ramakrishnan (Virginia Tech)
Spiros Papadimitriou (Rutgers University)
Koen Smets (University of Antwerp)
Hanghang Tong (CUNY)
Ye Wang (The Ohio State University)
Arthur Zimek (LMU Munich)

Preface

Traditionally, outlier mining and anomaly discovery focused on the automatic detection of highly deviating objects. It has been studied for several decades in statistics, machine learning, and data mining, and has led to a lot of insight as well as automated systems for the *detection* of outliers.

However, for today's applications to be successful, mere identification of anomalies alone is not enough. With more and more applications using outlier analysis for data exploration and knowledge discovery, the demand for manual verification and understanding of outliers is steadily increasing. Examples include applications such as health surveillance, fraud analysis, or sensor monitoring, where one is particularly interested in *why* an object seems outlying.

> Consider outlier analysis in the domain of health surveillance. An outlier might be a patient that shows high deviation in specific vital signals like “heart beat rate” and “skin humidity”. In case of health surveillance, only being *detected* as done by traditional algorithms is not sufficient: health professionals have to be able to verify the reasons for *why* this patient stands out in order to provide medical treatment accordingly. A key task in outlier analysis is to assist the expert in this verification. Hence, outlier mining algorithms should provide additional *descriptive information*. In particular, these outlier descriptions should highlight the specific deviation of an outlier in contrast to regular patients in easily understandable terms.

Even though outlier detection has been studied for several decades, awareness for the need of outlier descriptions is only recent. Mining outlier descriptions is currently being studied in different forms in different fields, such as in contrast mining, pattern mining, data compression, graph outlier mining, subspace outlier mining, in addition to other fields including data visualization, image saliency detection, and astronomy. We see a large overlap in the techniques of these different fields and believe the developments in either setting can have a significant impact on the other.

Our aim is to bring these and other communities together in one venue. With ODD, our objectives are to: 1) increase the general interest on this important topic in the broader research community; 2) bring together experts from closely related areas (e.g., outlier detection and contrast mining) to shed light on how this emerging research direction can benefit from other well-established areas; 3) provide a venue for active researchers to exchange ideas and explore important research issues in this area. Overall, the idea behind ODD is that outlier detection and description together will provide novel techniques that assist humans in manual outlier verification by easy-to-understand descriptions, and so will help to advance the state of the art and applicability of outlier mining.

The main program of ODD’13 consists of six research papers, selected out of nine submissions, together covering various aspects of outlier detection and description. We sincerely thank the authors of the submissions as well as the attendees of the workshop. We wish to thank the members of our program committee for their help in selecting a set of high-quality papers. Furthermore, we are very grateful to Charu Aggarwal and Raymond Ng for giving keynote presentations about their recent work on outlier analysis.

Leman Akoglu, Emmanuel Müller, Jilles Vreeken

July 2013

Table of Contents

Proceedings of the
ACM SIGKDD Workshop on
Outlier Detection and Description

These proceedings are also included in the ACM Digital Library.

ODD'13, August 11, 2013, Chicago, IL, USA

ACM SIGKDD Workshop on Outlier Detection and Description

General Chairs

Leman Akoglu (Stony Brook University)
Emmanuel Müller (Karlsruhe Institute of Technology)
Jilles Vreeken (Universiteit Antwerpen)

Program Committee

Fabrizio Angiulli (University of Calabria)
Ira Assent (Aarhus University)
James Bailey (University of Melbourne)
Arindam Banerjee (University of Minnesota)
Albert Bifet (Yahoo! Labs Barcelona)
Christian Böhm (LMU Munich)
Rajmonda Caceres (MIT)
Varun Chandola (Oak Ridge National Laboratory)
Polo Chau (Georgia Tech)
Sanjay Chawla (University of Sydney)
Tijl De Bie (University of Bristol)
Christos Faloutsos (Carnegie Mellon University)
Jing Gao (University of Buffalo)
Manish Gupta (Microsoft (India)
Jaakko Holmén (Aalto University)
Eamonn Keogh (University of California – Riverside)
Matthijs van Leeuwen (KU Leuven)
Daniel B. Neill (Carnegie Mellon University)
Naren Ramakrishnan (Virginia Tech)
Spiros Papadimitriou (Rutgers University)
Koen Smets (University of Antwerp)
Hanghang Tong (CUNY)
Ye Wang (The Ohio State University)
Arthur Zimek (LMU Munich)

Preface

Traditionally, outlier mining and anomaly discovery focused on the automatic detection of highly deviating objects. It has been studied for several decades in statistics, machine learning, and data mining, and has led to a lot of insight as well as automated systems for the *detection* of outliers.

However, for today's applications to be successful, mere identification of anomalies alone is not enough. With more and more applications using outlier analysis for data exploration and knowledge discovery, the demand for manual verification and understanding of outliers is steadily increasing. Examples include applications such as health surveillance, fraud analysis, or sensor monitoring, where one is particularly interested in *why* an object seems outlying.

> Consider outlier analysis in the domain of health surveillance. An outlier might be a patient that shows high deviation in specific vital signals like “heart beat rate” and “skin humidity”. In case of health surveillance, only being *detected* as done by traditional algorithms is not sufficient: health professionals have to be able to verify the reasons for *why* this patient stands out in order to provide medical treatment accordingly. A key task in outlier analysis is to assist the expert in this verification. Hence, outlier mining algorithms should provide additional *descriptive information*. In particular, these outlier descriptions should highlight the specific deviation of an outlier in contrast to regular patients in easily understandable terms.

Even though outlier detection has been studied for several decades, awareness for the need of outlier descriptions is only recent. Mining outlier descriptions is currently being studied in different forms in different fields, such as in contrast mining, pattern mining, data compression, graph outlier mining, subspace outlier mining, in addition to other fields including data visualization, image saliency detection, and astronomy. We see a large overlap in the techniques of these different fields and believe the developments in either setting can have a significant impact on the other.

Our aim is to bring these and other communities together in one venue. With ODD, our objectives are to: 1) increase the general interest on this important topic in the broader research community; 2) bring together experts from closely related areas (e.g., outlier detection and contrast mining) to shed light on how this emerging research direction can benefit from other well-established areas; 3) provide a venue for active researchers to exchange ideas and explore important research issues in this area. Overall, the idea behind ODD is that outlier detection and description together will provide novel techniques that assist humans in manual outlier verification by easy-to-understand descriptions, and so will help to advance the state of the art and applicability of outlier mining.

The main program of ODD’13 consists of six research papers, selected out of nine submissions, together covering various aspects of outlier detection and description. We sincerely thank the authors of the submissions as well as the attendees of the workshop. We wish to thank the members of our program committee for their help in selecting a set of high-quality papers. Furthermore, we are very grateful to Charu Aggarwal and Raymond Ng for giving keynote presentations about their recent work on outlier analysis.

Leman Akoglu, Emmanuel Müller, Jilles Vreeken

July 2013

Table of Contents

Invited Talk

Outlier Detection in Personalized Medicine

Raymond Ng
Department of Computer Science
University of British Columbia, Vancouver, Canada
rng@cs.ubc.ca

Abstract

Personalized medicine has been hailed as one of the main directions for medical research in this century. In the first half of the talk, we give an overview on our personalized medicine projects that use gene expression, proteomics, DNA and clinical features. In the second half, we give two applications where outlier detection is valuable for the success of our work. The first one focuses on identifying mislabeled patients, and the second one deals with quality control of microarrays.

Bio

Dr. Raymond Ng is a professor in Computer Science at the University of British Columbia. His main research area for the past two decades is on data mining, with a specific focus on health informatics and text mining. He has published over 150 peer-reviewed publications on data clustering, outlier detection, OLAP processing, health informatics and text mining. He is the recipient of two best paper awards - from 2001 ACM SIGKDD conference and the 2005 ACM SIGMOD conference. He was one of the program co-chairs of the 2009 International conference on Data Engineering, and one of the program co-chairs of the 2002 ACM SIGKDD conference. He was also one of the general co-chairs of the 2008 ACM SIGMOD conference. He was an editorial board member of the Very large Database Journal and the IEEE Transactions on Knowledge and Data Engineering until 2008.

For the past decade, Dr. Ng has co-led several large scale genomic projects, funded by Genome Canada, Genome BC and NSERC. The total amount of funding of those projects well exceeded $40 million Canadian dollars. He now holds the Chief Informatics Officer position of the PROOF Centre of Excellence, which focuses on biomarker development for end-stage organ failures.

ODD'13, August 11, 2013, Chicago, IL, USA.
ACM 978-1-4503-2335-2.

Enhancing One-class Support Vector Machines for Unsupervised Anomaly Detection

Mennatallah Amer
Department of Computer Science and Engineering
German University in Cairo, Egypt
mennatallah.amer@student.guc.edu.eg

Markus Goldstein
German Research Center for Artificial Intelligence (DFKI GmbH)
D-67663 Kaiserslautern, Germany
Markus.Goldstein@dfki.de

Slim Abdennadher
Department of Computer Science and Engineering
German University in Cairo, Egypt
slim.abdennadher@guc.edu.eg

ABSTRACT

Support Vector Machines (SVMs) have been one of the most successful machine learning techniques for the past decade. For anomaly detection, also a semi-supervised variant, the one-class SVM, exists. Here, only normal data is required for training before anomalies can be detected. In theory, the one-class SVM could also be used in an unsupervised anomaly detection setup, where no prior training is conducted. Unfortunately, it turns out that a one-class SVM is sensitive to outliers in the data. In this work, we apply two modifications in order to make one-class SVMs more suitable for unsupervised anomaly detection: Robust one-class SVMs and eta one-class SVMs. The key idea of both modifications is, that outliers should contribute less to the decision boundary as normal instances. Experiments performed on datasets from UCI machine learning repository show that our modifications are very promising: Comparing with other standard unsupervised anomaly detection algorithms, the enhanced one-class SVMs are superior on two out of four datasets. In particular, the proposed eta one-class SVM has shown the most promising results.

Keywords

One-Class SVM, Outlier Detection, Outlier Score, Support Vector Machines, Unsupervised Anomaly Detection

1. INTRODUCTION

Anomalies or outliers are instances in a dataset, which deviate from the majority of the data. Anomaly detection is the task of successfully identifying those records within a given dataset. Applications that utilize anomaly detection include intrusion detection [22], medical diagnosis [17], fraud detection [29] and surveillance [3].

In the anomaly detection domain, three different learning setups based on the availability of labels exist [7]: Similar to standard classification tasks, a supervised learning approach can be used to detect anomalies. In this case, a training dataset containing normal and outlying instances, which is used to learn a model. The learned model is then applied on the test dataset in order to classify unlabeled records into normal and anomalous records. The second learning approach is semi-supervised, where the algorithm models the normal records only. Records that do not comply with this model are labeled as outliers in the testing phase. The last learning setup is unsupervised. Here, the data does not contain any labeling information and no separation into a training and testing phase is given. Unsupervised learning algorithms assume that only a small fraction of the data is outlying and that the outliers exhibit a significantly different behavior than the normal records.

In many practical application domains, the unsupervised learning approach is particularly suited when no labeling information is available. Moreover, in some applications the nature of the anomalous records is constantly changing, thus obtaining a training dataset that accurately describe outliers is almost impossible. On the other hand, unsupervised anomaly detection is the most difficult setup since there is no decision boundary to learn and the decision is only based on intrinsic information of the dataset.

Unsupervised anomaly detection algorithms can be categorized according to their basic underlying methodology [7]. The most popular and also often best performing category for unsupervised learning are nearest-neighbor based methods. The strength of those algorithms stem from the fact that they are inherently unsupervised and have an intuitive criteria for detecting outliers. Their limitations include the quadratic computational complexity and a possible incorrectness when handling high dimensional data.

Support Vector Machines are today a very popular machine learning technique that can be used in a variety of applications. This includes for example handwritten digit recognition, object recognition, speaker identification, text categorization [6] and also anomaly detection. In those applications, SVMs perform at least as good as other methods in terms of the generalization error [6]. SVMs take the capacity of the model into account, which is the flexibility of the learned model to represent any training dataset with a minimal error. This makes SVMs a Structure Risk Minimization (SRM) procedure which is a stimulating alternative to the traditional Empirical Risk Minimization (ERM) procedures.

ODD'13, August 11th, 2013, Chicago, IL, USA.

There are many factors that contributed to the high popularity of SVMs today. First of all, its theory is heavily investigated and it comes with a convex optimization objective ensuring that the global optimum will be reached. Moreover, its solution is sparse making it really efficient in comparison to other kernel-based approaches [4]. Finally, some kernels even allow SVMs to be considered as a dimensionality reduction technique [32]. Thus it is argued that it can be used to overcome the "curse of dimensionality", which make SVMs theoretically very attractive for the unsupervised anomaly detection problem.

2. RELATED WORK

As already mentioned, the most popular category for unsupervised anomaly detection are nearest-neighbor based algorithms. Here, global methods, for example the k-nearest neighbor [23, 2] and local methods exist. For the latter a huge variety of algorithms have been developed, many based on the Local Outlier Factor (LOF) [5]: the Connectivity-Based Outlier Factor (COF) [27], the Local Outlier Probability (LoOP) [15], the Influenced Outlierness (INFLO) [14] and the parameter-free Local Correlation Integral (LOCI) [21]. All basically assume that outliers lie in sparse neighborhoods and are far away from their nearest-neighbors [7].

Clustering based algorithms cluster the data and measure the distance from each instance to its nearest cluster center. The basic assumption is that outliers are far away from the normal clusters or appear in small clusters [7]. Algorithms include the Cluster-based Outlier Factor (CBLOF) [12] and the Local Density Cluster-based Outlier Factor (LDCOF) [1]. For the unsupervised anomaly detection problem, the nearest-neighbor based algorithms tend to be more capable of accurately identifying outliers [1]. On the other hand, clustering based anomaly detection has theoretically a lower computational effort, such that it could be preferred in cases where large datasets have to be processed.

Among these two often used categories, also others have been investigated: Classification algorithms, statistical approaches, Artificial Neural Networks (ANNs) and Support Vector Machine (SVMs) [7]. The majority of these categories require a labeled training set and hence they are of little applicability in an unsupervised learning setting. The Histogram-based Outlier Score (HBOS) is an unsupervised statistical based approach that was suggested in [10]. It computes a histogram for each feature individually and then the univariate results are combined in order to produce the final score. It is significantly faster than the other unsupervised anomaly detection algorithms at the expense of precision. Replicator Neural Networks (RNNs) [11] are a semi-supervised neural network based approach. Here, an artificial neural network is trained such that the output is a replica of the input. The reconstruction error is then used as an anomaly score. Another semi-supervised approach is the one-class SVM [25], a special variant of a SVM that is used for novelty detection. Details of which are covered in Section 3. However, a one-class SVM could also be used in an unsupervised setup. Then, training and testing is applied on the same data. Unfortunately, the training on a dataset already containing anomalies does not result in a good model. This is due to the fact that outliers can influence the decision boundary of a one-class SVM significantly.

In a supervised anomaly detection setting, Mukkamala et al. [20] showed that SVM based algorithms are superior compared to ANN based algorithms for the intrusion detection problem. SVMs had a shorter training time and produced better accuracy. The authors stated that the main limitation of SVMs is the fact that it is a binary classifier only. This limits the breadth of information that can be obtained about the type and degree of intrusions.

One class classification (OCC) is the task of learning to describe a target class in order to effectively identify its members. Following Vapnik's [31] intuition, most approaches attempt to find a boundary around the dataset. The formulation of one-class SVM proposed by Schölkopf et al [25] finds the boundary in the form a hyperplane. This is the formulation that we attempt to enhance. Support vector domain description (SVDD) proposed by Tax et al. [28] strives to find the minimum enclosing hypersphere that best describes the data. Both of the above mentioned formulations produce an equivalent solution in case of constant kernel diagonal entries [25]. Quarter-sphere support vector machines [16] were designed to handle intrusion detection data which have one-sided features centered around the origin. It fixes the center of the quarter sphere at the origin yielding a much simpler linear programming optimization objective. Liu and Zheng [18] proposed a SVM variant called MEMEM that combines between the discriminative capabilities of SVMs and the descriptive capabilities of one-class SVMs. This makes it particularly suited for handling unbalanced datasets. However it is a completely supervised approach.

3. ONE-CLASS SVMs

3.1 Motivation

In contrast to traditional SVMs, one-class SVMs attempt to learn a decision boundary that achieves the maximum separation between the points and the origin [24]. Interestingly this was the initial idea from which traditional supervised SVMs emerged. Its origin date back to the earliest work of Vapnik et al. in 1963 [30]. The idea was hindered by the inability to learn non-linear decision boundaries as well as the inability to account for outliers. Both of these problems were solved by the introduction of kernels and the incorporation of soft margins. A one-class SVM uses an implicit transformation function $\phi(\cdot)$ defined by the kernel to project the data into a higher dimensional space. The algorithm then learns the decision boundary (a hyperplane) that separates the majority of the data from the origin. Only a small fraction of data points are allowed to lie on the other side of the decision boundary: Those data points are considered as outliers.

The Gaussian kernel in particular guarantees the existence of such a decision boundary [24]. By observing that all the kernel entries are non-negative, it can be concluded that all the data in the kernel space lies in the same quadrant. This makes the Gaussian kernel well suited to deal with any arbitrary dataset. Let the function $g(\cdot)$ be defined as follows:

$$g(x) = w^T \phi(x) - \rho \tag{1}$$

where w is the vector perpendicular to the decision boundary and ρ is the bias term. Then, Equation 2 shows the decision function that one-class SVMs use in order to identify normal points. The function returns a positive value for normal

points, negative otherwise:

$$f(x) = sgn(g(x)). \tag{2}$$

One-class SVMs are traditionally used in a semi-supervised setting. The output of the algorithm is a binary label specifying whether the point is normal or not.

3.2 Objective

Equation 3 shows the primary objective of one-class SVMs:

$$\begin{aligned} &\min_{w,\xi,\rho} \frac{\|w\|^2}{2} - \rho + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i \\ &\text{subject to: } w^T\phi(x_i) \geq \rho - \xi_i,\ \xi_i \geq 0, \end{aligned} \tag{3}$$

where ξ_i is the slack variable for point i that allows it to lie on the other side of the decision boundary, n is the size of the training dataset and ν is the regularization parameter.

The deduction from the theoretical to the mathematical objective can be stated by the distance to the decision boundary. The decision boundary is defined as:

$$g(x) = 0. \tag{4}$$

In this context, the distance of any arbitrary data point to the decision boundary can be computed as:

$$d(x) = \frac{|g(x)|}{\|w\|}. \tag{5}$$

Thus, the distance that the algorithm attempts to maximize can be obtained by plugging the origin into the equation yielding $\frac{\rho}{\|w\|}$. This can also be stated as the minimization of $\frac{\|w\|^2}{2} - \rho$.

The second part of the primary objective is the minimization of the slack variables ξ_i for all points. ν is the regularization parameter and it represents an upper bound on the fraction of outliers and a lower bound on the number of support vectors. Varying ν controls the trade-off between ξ and ρ.

To this end, the primary objective is transformed into a dual objective, shown in Equation 6. The transformation allows SVMs to utilize the kernel trick as well as to reduce the number of variables to one vector. It basically yields a Quadratic Programming (QP) optimization objective.

$$\begin{aligned} &\min_{\alpha} \frac{\alpha^T Q \alpha}{2} \\ &\text{subject to: } 0 \leq \alpha_i \leq \frac{1}{\nu n}, \sum_{i=1}^{n}\alpha_i = 1, \end{aligned} \tag{6}$$

where Q is the kernel matrix and α are the Lagrange multipliers.

3.3 Outlier Score

A continuous outlier score reveals more information than a simple binary label such as the output of Equation 2. Similar to [1], our goal is to compute an anomaly score such that a larger score corresponds to significantly outlying points.

In Equation 7, we propose a possible way to compute such a score. Here, g_{max} is the maximum directed distance between the dataset points and the decision boundary. The score is scaled by that distance such that the points that are lying on the decision boundary would have an outlier score of 1.0 similar to [5]. A score larger than 1.0 indicates that the point is a potential outlier.

$$f(x) = \frac{g_{max} - g(x)}{g_{max}} \tag{7}$$

3.4 Influence of Outliers

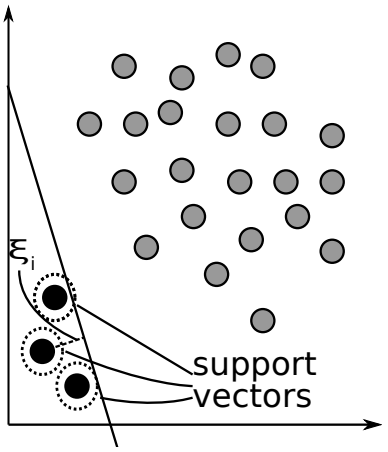


Figure 1: A 2 dimensional example of the decision boundary in the kernel space learned by a one-class SVM.

Figure 1 shows an example of a resulting decision boundary in the presence of outliers in the dataset. The decision boundary is shifted towards the outlying black points and they are additionally the support vectors in this example. Thus the outliers are the main contributors to the shape of the decision boundary. Whilst the shifting of the decision boundary might not have a great influence on the overall rank of the points when using Equation 7, the shape of the decision boundary will. To overcome this problem, in the following section two methods are proposed in order to make the decision boundary less dependent on these outliers.

4. ENHANCING ONE-CLASS SVMs

In this section two approaches are proposed to tackle the challenge that outliers do significantly contribute to the decision boundary. Both approaches are inspired from work done in order to make traditional supervised SVMs more robust against noise in the training dataset. They have the additional advantage of maintaining the sparsity of the SVM solution.

4.1 Robust One-class SVMs

4.1.1 Motivation

The approach is based on Song et al. [26], where the authors attempted to make the supervised SVM more robust in case of existing outliers in the training data. The key idea is the minimization of the of the Mean Square Error (MSE) for tackling outliers using the center of class as an averaged information. The conducted experiments showed that the generalization performance improved and the number of support vector decreased compared to the traditional SVM.

The main modification of robust one-class SVMs is with respect to the slack variables. As illustrated in Figure 1, a non-zero slack variable ξ_i allows a point x_i to lie on the other side of the decision boundary. In the case of robust one-class SVMs, the slack variables are proportional to the distance to the centroid. This allows points that are distant from the center to have a large slack variable. Since the slack variables are fixed, they are dropped from the minimization objective. On the one hand, this causes the decision boundary to be shifted towards the normal points. On the other hand, it loses part of the interpretability of the results as there is no restriction on the number of points that can appear on the other side of the decision boundary. Theoretically, all the points can be labeled as outlying using Equation 2 and consequentially, the majority could have a score greater than 1.0 when using Equation 7.

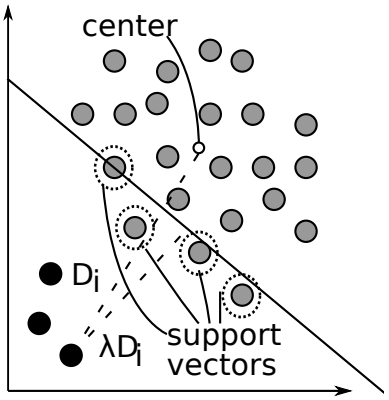


Figure 2: Modifying the slack variables for robust one-class SVMs. Each slack variable is proportional to the distance to the centroid. Dropping the minimization of the slack variables from the objective function causes the decision boundary to be shifted towards the normal points.

Figure 2 illustrates how the slack variables are modified. Points that are further away from the center of the data are allowed to have a larger slack variable. Then, the decision boundary is shifted towards the normal points and the outliers are no longer support vectors.

4.1.2 Objective

The objective of the proposed robust one-class SVMs is stated in Equation 8. Here, the slack variables are dropped from the minimization objective. They only appear in the constraints as $\hat{D}_i$, whereas λ is the regularization parameter.

$$\begin{aligned} &\min_{w,\rho} \frac{\|w\|^2}{2} - \rho \\ &\text{subject to } w^T\phi(x_i) \geq \rho - \lambda * \hat{D}_i \end{aligned} \tag{8}$$

The slack variable D_i is computed using Equation 9. It represents the distance to the centroid in the kernel space. Since the transformation function is implicitly defined by the kernel (Q), Equation 9 can not directly be used. Thus, an approximation that was introduced by Hu et al [13] is computed instead. This approximation is summarized in Equation 10. Here, the expression $\frac{1}{n}\sum_{i=1}^{n}\phi(x_i)\frac{1}{n}\sum_{i=1}^{n}\phi(x_i)$ is a constant and hence it can be dropped. The normalized distance $\hat{D}_i$ appears in the optimization Objective 8.

$$\begin{aligned} D_i &= \|\phi(x_i) - \frac{1}{n}\sum_{i=1}^{n}\phi(x_i)\|^2 \\ \hat{D}_i &= \frac{D_i}{D_{max}} \end{aligned} \tag{9}$$

$$\begin{aligned} D_i &= \|\phi(x_i) - \frac{1}{n}\sum_{i=1}^{n}\phi(x_i)\|^2 \\ &= Q(x_i,x_i) - \frac{2}{n}\sum_{j=1}^{n}Q(x_i,x_j) - \frac{1}{n}\sum_{i=1}^{n}\phi(x_i)\frac{1}{n}\sum_{i=1}^{n}\phi(x_i) \\ &\approx Q(x_i,x_i) - \frac{2}{n}\sum_{j=1}^{n}Q(x_i,x_j) \end{aligned} \tag{10}$$

The dual objective of the robust one-class SVM can be summarized as follows:

$$\begin{aligned} &\min_{\alpha} \frac{\alpha^T Q\alpha}{2} + \lambda D^T\alpha \\ &\text{subject to } 0 \leq \alpha \leq 1, e^T\alpha = 1 \end{aligned} \tag{11}$$

It can be seen that it is only a minor modification to the dual objective of the one-class SVM objective in Equation 6 and hence it can be incorporated easily in the original solver.

4.2 Eta One-class SVMs

4.2.1 Motivation

In contrast to robust one-class SVMs, this approach uses an explicit outlier suppression mechanism. The methodology for supervised SVMs was first proposed by Xu et al. [33]. This suppression mechanism is achieved by introducing a variable η, which represents an estimate that a point is normal. Thus an outlying point would ideally have η set to zero. This variable controls the portion of the slack variables that is going to contribute to the minimization objective.

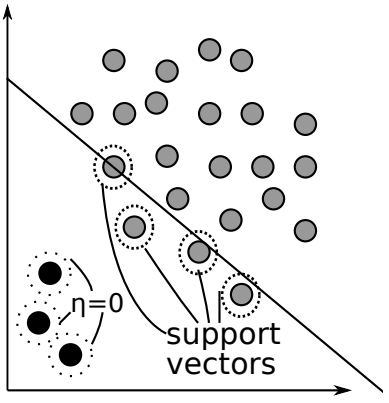


Figure 3: The idea of the eta one-class SVM: Outliers have small values for η and do thus not contribute to the decision boundary.

Figure 3 shows how the introduction of η affects the decision boundary. The outlying points would be assigned $\eta = 0$ thus they would not be considered whilst learning the decision boundary. Here, the decision boundary would be influenced only by the normal points.

4.2.2 Objective

Equation 12 shows the objective of the eta one-class SVM. Outlying points would have η set to 0 and hence they would not be contributing to the optimization objective. The disadvantage of introducing η is that the objective loses part of its intuitive interpretation: Minimizing the slack variable is equivalent to minimizing the number of outliers. A variable β is introduced in order to cope with this challenge. It controls the maximum number of points that are allowed to be outlying:

$$\min_{w,\rho} \min_{\eta_i \in \{0,1\}} \frac{\|w\|^2}{2} - \rho + \sum_{i=1}^{n} \eta_i max(0, \rho - w^T * \phi(x_i)), \\ \text{subject to } e^T \eta \geq \beta n. \tag{12}$$

The objective is composed of two parts: A convex quadratic problem in w for a fixed η, and a linear problem in η for a fixed w. However, the objective is not jointly convex. This means that minimizing each part alternatively is not guaranteed to yield a global minimum. The above formulation will be relaxed similar to what was proposed in the original work [33] into a semi-definite problem. Then, it will be relaxed into a iterative formulation due to the limited practicability of semi-definite programs. The iterative relaxation is achieved using concave duality similar to what was used by Zhou et al. [36].

Semi-Definite Programming Problem

The non-convex optimization objective of Equation 12 can be relaxed by relaxing the constraints on η. For a fixed η, introducing Lagrange multipliers would yield the following dual objective:

$$\min_{0 \leq \eta \leq 1, M = \eta * \eta^T} \max_{0 \leq \alpha \leq 1} \frac{\alpha^T Q \cdot M \alpha}{2}, \\ \text{subject to } e^T * \eta \geq \beta n,\ \alpha^T \eta = 1,\ 0 \leq \alpha \leq 1. \tag{13}$$

The formulation in Equation 13 is convex in both, η and α. The final obstacle is the constraint on matrix M as it is a non-convex quadratic constraint. The constraint can be approximated to $M \succeq \eta * \eta^T$ yielding a convex optimization objective:

$$\min_{0 \leq \eta \leq 1} \min_{M \succeq \eta * \eta^T} \max_{0 \leq \alpha \leq 1} \frac{\alpha^T Q \cdot M \alpha}{2} \tag{14}$$

Objective 14 is equivalent to solving the following semidefinite programming (SDP) problem:

$$\min_{\eta, \delta, \gamma, \sigma, M} \delta \\ \text{subject to } e^T \eta \geq \beta n, 0 \leq \eta \leq 1, \gamma \geq 0, \sigma \geq 0, \\ M \succeq \eta * \eta^T \\ \begin{bmatrix} 2*(\delta - e^T * \sigma) & (\gamma - \sigma)^T \\ \gamma - \sigma & Q \cdot M \end{bmatrix} \succeq 0 \\ \begin{bmatrix} 1 & \eta^T \\ \gamma - \sigma & Q \cdot M \end{bmatrix} = 0. \tag{15}$$

Iterative Relaxation

The SDP solution is expensive to compute and hence an alternative approach was proposed by Zhou et al. [36]. It uses a concave duality in order to relax Equation 12 into a multi-stage iterative problem. A discussion of why the procedure yields a good approximation is given by Zhang [35]. The relaxation yields an objective that has a convex and a concave part, which makes the iterative approach a generalization of a concave convex procedure (OCCC) [34] that is guaranteed to converge.

Let the non-convex regularization in Equation 12 correspond to $g(h(w))$, where $h(w) = max(0, \rho - w^T \phi(x))$ and $g(u) = \inf_{\eta \in \{0,1\}} [\eta^T u]$, using concave duality, the objective can be reformulated into

$$\min_{w,\rho,\eta} E_{vex} + E_{cave} \\ E_{vex} = \frac{\|w\|^2}{2} - \rho + \eta^T h(w),\ E_{cave} = g^*(\eta), \tag{16}$$

where g^* is the concave dual of g.

Equation 16 can be solved by iteratively minimizing E_{vex} and E_{cave}. Initially η is set to a vector of ones. Then the following steps are done until convergence:

1. For a fixed η, minimize E_{vex} which corresponds to the following dual objective:

$$\min_{\alpha} \frac{\alpha^T Q \cdot N \alpha}{2}, \\ \text{where } N = \eta * \eta^T, \\ \text{subject to } \alpha^T \eta = 1,\ 0 \leq \alpha \leq 1.$$

2. For fixed w and ρ, the minimum of E_{cave} is at:

$$u_i = max(0, \rho - w^T \phi(x_i)), \\ \eta_i = I(\beta n - s(i))$$

where $s(i)$ is the order of function over u arranged in ascending order and I is the indicator function.

5. EXPERIMENTS

In this section, all the proposed one-class SVM based algorithms are compared against standard nearest-neighbor, clustering and statistical based unsupervised anomaly detection algorithms. The experiments were conducted using RapidMiner [19], where all of the algorithms are implemented in the Anomaly Detection extension[1]. The SVM based algorithms are all using the Gaussian kernel, the spread of the kernel was tuned similar to what is proposed by Evangelista et al. [8]. For the Gaussian Kernel, it is desirable to attain diverse kernel entries as it is a measure of similarity between data points. Evangelista et al. achieved that by maximizing the ratio of the standard deviation of the non-diagonal entries of the kernel matrix to its mean. The maximization objective is solved using gradient ascent.

The area under the ROC curve (AUC) is used as a performance measure, where the curve is created by varying the outlier threshold. It basically measures the quality of the ranking of outliers among normal records. The results of the AUC of running the different algorithms are included in Table 4. Figure 4 shows exemplary ROC curves of the algorithms for two different datasets. In each subfigure, the three SVM based algorithms are compared against the best performing algorithm from each of the other categories.

Another important comparison is between the standard semi-supervised one-class SVM and the proposed improvements of this work: the robust one-class SVM and eta one-class SVM. In addition to the performance measured by the

[1]Available at `http://code.google.com/p/rapidminer-anomalydetection/`

AUC, also the number of support vectors is an important factor to consider as it directly affects the computation time of the SVM based algorithms. The number of support vectors are shown in Table 2. The average CPU execution time of the algorithms over 10 runs is shown in Table 3.

5.1 Datasets

Datasets from the UCI machine learning repository [9] are used for the evaluation of the anomaly detection algorithms. Most of the datasets of UCI repository are traditionally dedicated for classification tasks. Hence they have to be preprocessed in order to serve for the evaluation of unsupervised anomaly detection algorithms. This is typically performed by picking a meaningful outlying class and sampling the outliers to a small fraction [1]. Table 1 summarizes the characteristics of the preprocessed datasets.

The preprocessing was also performed using RapidMiner. For *ionosphere*, *shuttle* and *satellite*, stratified sampling was used to reduce the number of outliers (for reproducibility, the pseudo random generator seed was set to 1992). The preprocessing of the *breast-cancer* dataset was identical to the one proposed in [15].

5.2 Results

The results of the *shuttle* dataset are shown in Figure 4(a). Here, the eta one-class SVM is superior to all the other algorithms. The statistical based algorithm Histogram outperforms the nearest-neighbor and clustering based algorithms. It also outperforms the robust one-class SVM. Surprisingly, the standard one-class SVM outperforms the robust one-class SVM for the shuttle dataset. However, robust one-class produces a much sparser solution with only 5 support vectors dropping the CPU execution by two thirds. Figure 4(b) illustrates the results of the *satellite* dataset. Here, the SVM based algorithms performed worst among all existing categories. The performance of the algorithms is comparable at the first portion of the dataset. Which means that they perform equally well in predicting the top outliers.

Table 4 summarizes the results in terms of AUC for all algorithms on all four datasets. It can be seen that all SVM based algorithms perform generally well on all datasets. For *ionosphere* and *shuttle* the eta one-class SVM is even superior. For the *breast-cancer* dataset, SVM based algorithms score on average. For the satellite dataset, where also many support vectors have been found, results are below the average.

Table 2: Number of support vectors of SVM based algorithms

Algorithm	*ionosphere*	*shuttle*	*breast-cancer*	*satellite*
One-class	106	21374	144	2085
Robust One-class	116	5	90	385
Eta One-class	37	8	48	158

6. DISCUSSION AND CONCLUSION

The experiments showed that the proposed SVM based algorithms are well suited for the unsupervised anomaly detection problem. In two out of four datasets, SVM based algorithms are even superior. They constantly outperform all clustering based algorithms. In general, they perform at least average on unsupervised anomaly detection problems. For the *satellite* dataset, the performance of the SVM based algorithms is slightly below the average. The main reason why this is a challenging dataset for SVM based algorithms is not known exactly, but we can observe that in this case the number of support vectors is comparably high.

When comparing the SVM based algorithms with each other, the eta one-class SVM seems to be the most promising one. On average, it produces a sparse solution and it also performs best in terms of AUC. In general, the robust one-class SVM produces a sparser solution than the standard one-class SVM, but in term of performance, there is no significant improvement. In terms of time efficiency, for larger datasets the enhanced algorithms are more efficient due to the sparsity property.

When looking at computational effort, SVM based algorithms have in general less than a quadratic time complexity due to the sparsity property. However, the parameter tuning for the Gaussian kernel similar to [8] pushes the complexity back to quadratic time.

Additionally, we introduced a method for calculating an outlier score based on the distance to the decision boundary. In contrast to the binary label assigned by standard one-class SVMs, it allows to rank the outliers, which is often essential in an unsupervised anomaly detection setup. The score also has a reference point, which means that scores in the range of [0,1] can be considered to be normal.

In conclusion, SVM based algorithms have shown that they can perform reasonably well for unsupervised anomaly detection. Especially the eta one-class SVM is a suitable candidate for investigation when applying unsupervised anomaly detection in practice.

Acknowledgment

This work was partially funded by the Rheinland-Palatinate Foundation for Innovation, project AnDruDok (961-38 6261 / 1039).

7. REFERENCES

[1] Mennatallah Amer and Markus Goldstein. Nearest-neighbor and clustering based anomaly detection algorithms for rapidminer. *Proc. of the 3rd RapidMiner Community Meeting and Conference (RCOMM 2012)*, pages 1–12, 2012.

[2] Fabrizio Angiulli and Clara Pizzuti. Fast outlier detection in high dimensional spaces. In Tapio Elomaa, Heikki Mannila, and Hannu Toivonen, editors, *Principles of Data Mining and Knowledge Discovery*, volume 2431 of *Lecture Notes in Computer Science*, pages 43–78. Springer, 2002.

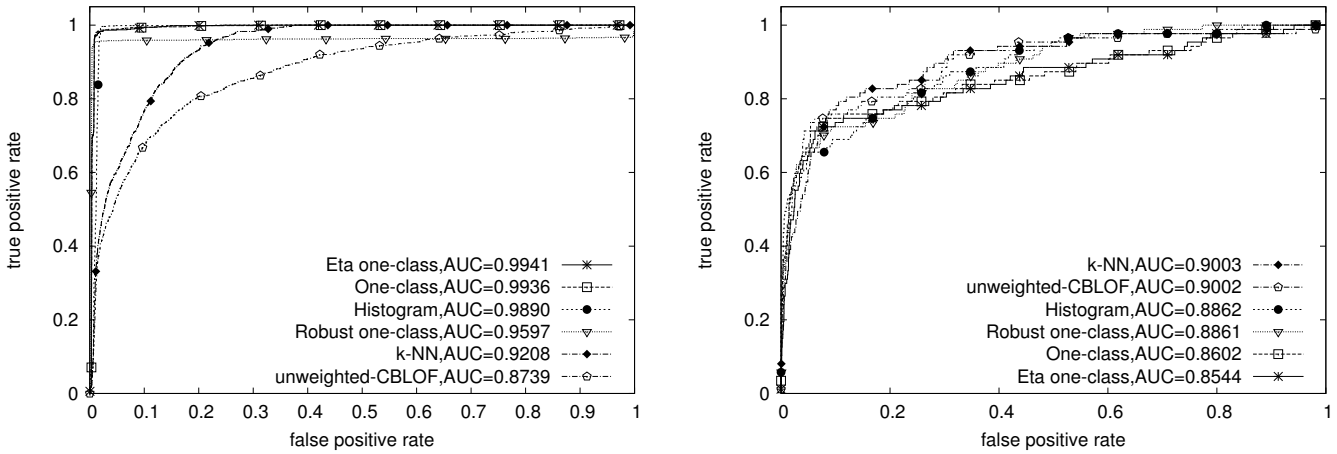
[3] Arslan Basharat, Alexei Gritai, and Mubarak Shah. Learning object motion patterns for anomaly detection and improved object detection. *CVPR*, pages 1–8, 01 2008.

[4] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 1st ed. 2006. corr. 2nd printing 2011 edition, October 2007.

[5] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. LOF: Identifying density-based local outliers. In *Proc. of the 2000 ACM SIGMOD Int. Conf. on Management of Data*, pages 93–104, Dallas, Texas, USA, 05 2000. ACM.

Table 1: Datasets used for evaluation. The preprocessing selects particular classes as outliers and samples it down to a small fraction in order to meet the requirements for unsupervised anomaly detection.

Meta-data	Original Size	Attributes	Outlier class(es)	Resulting dataset size	Sampled outliers percentage
ionosphere	351	26	b	233	3.4%
shuttle	58000	9	2 ,3 ,5 and 6	46464	1.89%
breast-cancer	569	30	M	367	2.72%
satellite	6435	36	2,4 and 5	4486	1.94%



(a) Shuttle (b) Satellite

Figure 4: ROC curves for SVM based algorithms and existing approaches. For the latter, the best performing algorithms of the categories nearest-neighbor, statistical and clustering based are plotted.

Table 3: CPU execution time of SVM based algorithms

Algorithm	*ionosphere* [ms]	*shuttle* [s]	*breast-cancer* [ms]	*satellite* [s]
One-class	21.55 ± 0.26	747.15 ± 10.94	48.72 ± 1.01	14.02 ± 2.00
Robust one-class	33.82 ± 0.26	218.93 ± 3.17	57.27 ± 2.29	8.60 ± 0.06
Eta one-class	27.48 ± 0.25	4.07 ± 0.14	82.46 ± 0.42	12.35 ± 0.95

Table 4: Comparing the AUC of SVM based algorithms against other anomaly detection algorithms

Dataset	One-class	Robust one-class	Eta one-class	k-NN	LOF	COF	INFLO	LoOP	Histogram	CBLOF	u-CBLOF	LDCOF
ionosphere	0.9878	0.9956	**0.9972**	0.9933	0.9178	0.9406	0.9406	0.9211	0.7489	0.3183	0.9822	0.9306
shuttle	0.9936	0.9597	**0.9941**	0.9208	0.6072	0.5612	0.5303	0.5655	0.9889	0.8700	0.8739	0.5312
breast-cancer	0.9843	0.9734	0.9833	0.9826	0.9916	0.9888	**0.9922**	0.9882	0.9829	0.8389	0.9743	0.9804
satellite	0.8602	0.8861	0.8544	**0.9003**	0.8964	0.8708	0.8592	0.8664	0.8862	0.4105	0.9002	0.8657

[6] Christopher J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

[7] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3):15:1–15:58, July 2009.

[8] Paul F. Evangelista, Mark J. Embrechts, and Boleslaw K. Szymanski. Some properties of the gaussian kernel for one class learning. In *Proc. of the 17th Int. Conf. on Artificial neural networks*, ICANN'07, pages 269–278. Springer, 2007.

[9] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[10] Markus Goldstein and Andreas Dengel. Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm. In Stefan Wölfl, editor, *KI-2012: Poster and Demo Track*, pages 59–63. Online, 9 2012.

[11] Simon Hawkins, Hongxing He, Graham Williams, and Rohan Baxter. Outlier detection using replicator neural networks. In *In Proc. of the Fifth Int. Conf. and Data Warehousing and Knowledge Discovery (DaWaK02*, pages 170–180, 2002.

[12] Zengyou He, Xiaofei Xu, and Shengchun Deng. Discovering cluster-based local outliers. *Pattern Recognition Letters*, 24(9-10):1641–1650, 2003.

[13] W.J. Hu and Q. Song. An accelerated decomposition algorithm for robust support vector machines. *Circuits and Systems II: Express Briefs, IEEE Transactions on*, 51(5):234–240, 2004.

[14] Wen Jin, Anthony Tung, Jiawei Han, and Wei Wang. Ranking outliers using symmetric neighborhood relationship. In Wee-Keong Ng and et al., editors,

Advances in Knowledge Discovery and Data Mining, volume 3918 of *Lecture Notes in Computer Science*, pages 577–593. Springer, 2006.

[15] Hans-Peter Kriegel, Peer Kröger, Erich Schubert, and Arthur Zimek. Loop: local outlier probabilities. In *Proceeding of the 18th ACM Conf. on Information and knowledge management*, CIKM '09, pages 1649–1652, New York, NY, USA, 2009. ACM.

[16] Pavel Laskov, Christin Schäfer, Igor V. Kotenko, and Klaus-Robert Müller. Intrusion detection in unlabeled data with quarter-sphere support vector machines. *Praxis der Informationsverarbeitung und Kommunikation*, 27(4):228–236, 2007.

[17] Jessica Lin, Eamonn Keogh, Ada Fu, and Helga Van Herle. Approximations to magic: Finding unusual medical time series. In *In 18th IEEE Symp. on Computer-Based Medical Systems (CBMS*, pages 23–24, 2005.

[18] Yi Liu and Yuan F. Zheng. Minimum enclosing and maximum excluding machine for pattern description and discrimination. *Pattern Recognition, International Conference on*, 3:129–132, 2006.

[19] I. Mierswa, M. Wurst, R. Klinkenberg, M. Scholz, and T. Euler. Yale (now: Rapidminer): Rapid prototyping for complex data mining tasks. In *Proc. of the ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD 2006)*, 2006.

[20] S. Mukkamala, G. Janoski, and A. Sung. Intrusion detection using neural networks and support vector machines. *Proc. of the 2002 Int. Joint Conf. on Neural Networks. IJCNN'02 (Cat. No.02CH37290)*, pages 1702–1707, 2002.

[21] Spiros Papadimitriou, Hiroyuki Kitagawa, Phillip B. Gibbons, and Christos Faloutsos. Loci: Fast outlier detection using the local correlation integral. *Data Engineering, Int. Conf. on*, 0:315, 2003.

[22] Leonid Portnoy, Eleazar Eskin, and Sal Stolfo. Intrusion detection with unlabeled data using clustering. In *In Proc. of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001)*, pages 5–8, 2001.

[23] Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. Efficient algorithms for mining outliers from large data sets. In *Proc. of the 2000 ACM SIGMOD Int. Conf. on Management of Data*, SIGMOD '00, pages 427–438, New York, NY, USA, 2000. ACM.

[24] B Schölkopf, J C Platt, J Shawe-Taylor, a J Smola, and R C Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–71, July 2001.

[25] Bernhard Schölkopf, Robert C. Williamson, Alex J. Smola, John Shawe-Taylor, and John C. Platt. Support vector method for novelty detection. In *Advances in Neural Information Processing Systems 12, (NIPS) Conf.*, pages 582–588. The MIT Press, 11 1999.

[26] Qing Song, Wenjie Hu, and Wenfang Xie. Robust support vector machine with bullet hole image classification. *Systems, Man and Cybernetics, Part C, IEEE Transactions on*, 32(4):440–448, 2002.

[27] Jian Tang, Zhixiang Chen, Ada Fu, and David Cheung. Enhancing effectiveness of outlier detections for low density patterns. In Ming-Syan Chen, Philip Yu, and Bing Liu, editors, *Advances in Knowledge Discovery and Data Mining*, volume 2336 of *Lecture Notes in Computer Science*, pages 535–548. Springer, 2002.

[28] David M. J. Tax and Robert P. W. Duin. Support vector domain description. *Pattern Recognition Letters*, 20:1191–1199, 1999.

[29] Joost van Beusekom and Faisal Shafait. Distortion measurement for automatic document verification. In *Proc. of the 11th Int. Conf. on Document Analysis and Recognition*. IEEE, 9 2011.

[30] V. Vapnik and A. Lerner. Pattern recognition using generalized portrait method. *Automation and Remote Control*, 24:774–780, 1963.

[31] Vladimir N. Vapnik. *Statistical learning theory*. Wiley, 1 edition, September 1998.

[32] Wenjian Wang, Zongben Xu, Weizhen Lu, and Xiaoyun Zhang. Determination of the spread parameter in the gaussian kernel for classification and regression. *Neurocomputing*, 55(3-4):643–663, October 2003.

[33] Linli Xu, K Crammer, and Dale Schuurmans. Robust support vector machine training via convex outlier ablation. *Proc. of the National Conf. On Artificial Intelligence*, pages 536–542, 2006.

[34] Alan Yuille and Anand Rangarajan. The concave-convex procedure (cccp. In *Advances in Neural Information Processing Systems 14*. MIT Press, 2002.

[35] Tong Zhang. Analysis of multi-stage convex relaxation for sparse regularization. *J. Mach. Learn. Res.*, 11:1081–1107, March 2010.

[36] Xi-chuan Zhou, Hai-bin Shen, and Jie-ping Ye. Integrating outlier filtering in large margin training. *Journal of Zhejiang University-Science C*, 12(5):362–370, May 2011.

Systematic Construction of Anomaly Detection Benchmarks from Real Data

Andrew F. Emmott, Shubhomoy Das, Thomas Dietterich, Alan Fern, Weng-Keen Wong
Oregon State University
School of EECS
Corvallis, Oregon, USA
{emmott,dassh,tgd,afern,wong}@eecs.oregonstate.edu

ABSTRACT

Research in anomaly detection suffers from a lack of realistic and publicly-available problem sets. This paper discusses what properties such problem sets should possess. It then introduces a methodology for transforming existing classification data sets into ground-truthed benchmark data sets for anomaly detection. The methodology produces data sets that vary along three important dimensions: (a) point difficulty, (b) relative frequency of anomalies, and (c) clusteredness. We apply our generated datasets to benchmark several popular anomaly detection algorithms under a range of different conditions.

1. INTRODUCTION

Anomaly detection is an important task in many real-world applications, such as identifying novel threats in computer security [15, 23, 16, 21], finding interesting data points in scientific data [26], and detecting broken sensors (and other problems) in data sets [7]. Although a wide variety of anomaly detection algorithms have been developed and applied to these tasks [20, 11, 5, 31], a shortcoming of most published work is that there is no standard methodology for comparing anomaly detection methods. Instead, most published work either addresses data sets from specific applications or else employs synthetic data. This leads to three problems. First, with an application-specific data set, there is no independent way to assess the difficulty of the anomaly detection problem based on a standard set of properties of the data. Second, an application-specific data set limits us to the single data set from the application—there is no way to generate new data sets (aside from sub-sampling) that may differ in controlled ways. Third, with synthetic data sets, there is no real-world validity to the anomalies, so it is difficult to judge whether algorithms that work well on such simulated data will actually work well in a real-world setting.

In this paper, we attempt to address these shortcomings. In particular, our main contribution is to present a methodology for creating families of anomaly detection problems from real-world data sets. We begin in Section 2 by discussing the properties that benchmark data sets should possess in order to support rigorous evaluations of anomaly detection algorithms. Then in Section 3 we present the methodology that we have developed to create data sets with those properties. Section 4 describes an experiment in which we apply the methodology to benchmark several of the leading anomaly detection algorithms. Section 5 discusses the results of the experiment, and Section 6 presents our conclusions and suggestions for future work.

ODD'13, August 11th, 2013, Chicago, IL, USA.

2. REQUIREMENTS FOR ANOMALY DETECTION BENCHMARKS

The most common goal of anomaly detection is to raise an alarm when anomalous observations are encountered, such as insider threats [17], cyber attacks [15, 23, 16, 21], machine component failures [27, 28, 1], sensor failures [7], novel astronomical phenomena [26], or the emergence of cancer cells in normal tissue [22, 10]. In all of these cases, the underlying goal is to detect observations that are *semantically distinct* from normal observations. By this, we mean that the process that is generating the anomalies is different from the process that is generating the normal data points.

The importance of the underlying semantics suggests the first three requirements for benchmark datasets.

Requirement 1: Normal data points should be drawn from a real-world generating process. Generating data sets from some assumed probability distribution (e.g., a multivariate Gaussian) risks not capturing *any* real-world processes. Instead, as the field has learned from many years of experience with benchmark problems, it is important that the problems reflect the idiosyncrasies of real domains.

Requirement 2: The anomalous data points should also be from a real-world process that is semantically distinct from the process generating the normal points. The anomalous points should not just be points in the tails of the "normal" distribution. See, for example, Glasser and Lindauer's synthetic anomaly generator [8].

Requirement 3: Many benchmark datasets are needed. If we employ only a small number of data sets, we risk developing algorithms that only work on those problems. Hence,

we need a large (and continually expanding) set of benchmark data sets to ensure generality and prevent overfitting.

Requirement 4: Benchmark datasets should be characterized in terms of well defined and meaningful problem dimensions that can be systematically varied. An important goal for benchmarking is to gain insight into the strengths and weaknesses of the various algorithms. Ideally, we should identify those dimensions along which anomaly detection problems might vary and then generate benchmark data sets that vary these dimensions in a controlled fashion.

There is currently no established set of problem dimensions for anomaly detection, and we expect this set to evolve with experience. Here we propose four such dimensions: (a) point difficulty, (b) relative frequency, (c) semantic variation, and (d) feature relevance/irrelevance. The remainder of this section describes these in more detail.

Point difficulty measures the "distance" of an anomalous data point from the normal data points. We propose a point difficulty metric based on an oracle that knows the true generating processes underlying the "normal" and "anomalous" points. Using this knowledge, we suppose that the oracle can compute the probability $P(y = \text{normal}|x)$ that a data point x was generated by the "normal" distribution. The larger this value is for an anomalous point x, the more difficult it will be for an anomaly detection algorithm to discover that x is anomalous. One aspect of applying anomaly detection in adversarial settings (e.g., intrusion detection or insider threat detection) is that the adversaries try to blend in to the distribution of normal points.

Relative frequency is the fraction of the incoming data points that are (true) anomalies. The behavior of anomaly detection algorithms often changes with the relative frequency. If anomalies are rare, then methods that pretend that all training points are "normal" and fit a model to them may do well. If anomalies are common, then methods that attempt to fit a model of the anomalies may do well. In most experiments in the literature, the anomalies have a relative frequency between 0.01 and 0.1, but some go as high as 0.3 [14].

Semantic Variation is a measure of the degree to which the anomalies are generated by more than one underlying process. In this paper, we employ a measure of *clusteredness* as a proxy for this. If the anomalies are tightly clustered, then some anomaly detection algorithms will fail. For example, methods based on measures of local probability density will conclude that tightly clustered anomalies have high local density and hence are not anomalous.

Feature Relevance/Irrelevance. In applications, many candidate features are often available. However, many anomaly detection methods do not provide good feature selection mechanisms. Benchmark data sets should systematically vary the set of features to manipulate both the power of the relevant features and the number of irrelevant or "noise" features.

3. METHODOLOGY

We have developed a methodology that achieves most of the requirements listed above. To achieve the first three requirements, we develop 4,369 benchmark data sets by transforming 19 data sets chosen from the UC Irvine repository [2]. For each data set, we separate its data (e.g., the classes of a classification problem) into two sets: "normal" and "anomalous". This ensures that these data points are generated by distinct real-world processes rather than from synthesized distributions. To develop a measure of point difficulty, we fit a kernel logistic regression classifier to all of the available "normal" and "anomalous" data. This gives us an approximation to the oracle estimate of $P(y = \text{normal}|x)$. We can then manipulate the point difficulty of a benchmark data set by sampling the "anomalous" data points according to their point difficulty. It is easy to manipulate the relative frequency by varying the number of "anomalous" data points to include. We vary the degree of semantic variation by selecting data points that are either close together or far apart according to a simple distance metric. Our current methodology does not vary the feature relevance/irrelevance. This dimension is challenging to manipulate in a realistic manner, and we will investigate it further in future work.

3.1 Selecting Data Sets

To ensure reproducibility of our experiments, we only worked with data sets from the UCI data repository [2]. We selected all data sets that match the following criteria:

- *task*: Classification (binary or multi-class) or Regression. No Time-Series.
- *instances*: At least 1000. No upper limit.
- *features*: No more than 200. No lower limit.
- *values*: Numeric only. Categorical features are ignored if present. No missing values, except where easily ignored.

To ensure objectivity, we applied this fixed set of criteria rather than choosing data sets based on how well particular anomaly detection algorithms performed or based on our intuitions about which data sets might be better suited to creating anomaly detection problems.

If necessary, each data set was sub-sampled to 10,000 instances (while maintaining the class proportions for classification problems). Each feature was normalized to have zero mean and unit sample variance. We avoid time series because the majority of existing anomaly detection methods are based on models intended for independent and identically distributed data rather than for structured data such as time series data.

The 19 selected sets (grouped into natural categories) are the following:

- *binary classification*: MAGIC Gamma Telescope, MiniBooNE Particle Identification, Skin Segmentation, Spambase
- *multi-class classification*: Steel Plates Faults, Gas Sensor Array Drift, Image Segmentation, Landsat Satellite, Letter Recognition, Optical Recognition of Handwritten Digits, Page Blocks, Shuttle, Waveform, Yeast
- *regression*: Abalone, Communities and Crime, Concrete Compressive Strength, Wine, Year Prediction

3.2 Defining Normal versus Anomalous Data Points

A central goal of our methodology is that the "normal" and "anomalous" points should be produced by semantically distinct processes. To achieve this, we did the following.

For Irvine data sets that were already binary classification problems, we choose one class as "normal" and the other as "anomalous". Note that there is some risk that the "anomalous" points will have low semantic variation, since they all belong to a single class.

For multi-class data sets, we partition the available classes into two sets with the goal of maximizing the difficulty of telling them apart. Our heuristic procedure begins by training a Random Forest [3] to solve the multi-class classification problem. Then we calculate the amount of confusion between each class. For each data point x_i, the Random Forest computes an estimate of $P(\hat{y}_i|x_i)$, the predicted probability that x_i belongs to class $\hat{y}_i$. We construct a confusion matrix C in which cell $C[j,k]$ contains the sum of $P(\hat{y}_i = k|x_i)$ for all x_i whose true class $y_i = j$. We then define a graph in which each node is a class and each edge (between two classes j and k) has a weight equal to $C[j,k] + C[k,j]$. This is the (unnormalized) probability that a data point in class j will be confused with a data point in class k. We then compute the maximum weight spanning tree of this (complete) graph to identify a graph of "most-confusable" relationships between pairs of classes. We then two-color this tree so that no adjacent nodes have the same color. The two colors define the two sets of points. This approximately maximizes the confusions between "normal" and "anomalous" data points and also tends to make both the "normal" and "anomalous" sets diverse, which increases semantic variation in both sets.

For regression data sets, we compute the median of the regression response and partition the data into two classes by thresholding on this value. To the extent that low versus high values of the response correspond to different generative processes, this will create a semantic distinction between the "normal" and the "anomalous" data points. Points near the median will exhibit less semantic distinction, and they will also have high point difficulty.

3.3 Computing Point Difficulty

After reformulating all 19 Irvine tasks as binary classification problems, we simulate an omniscient oracle by applying Kernel Logistic Regression (KLR [12, 30, 13]) to fit a conditional probability model $P(y|x)$ to the data. Anomalies are labeled with $y = 0$ and normal points as $y = 1$. We then compute the logistic response for each candidate anomaly data point. Observe that points that are easy to discern from the "normal" class will have responses $P(y = 1|x)$ tending toward 0, while points that KLR confuses with the "normal" class will have responses above 0.5. Hence, for anomalous points, this response gives us a good measure of point difficulty.

For purposes of generating data sets, we assign each "anomalous" data point to one of four difficulty categories:

- *easy*: Difficulty score $\in (0, 0.1\overline{6})$
- *medium*: Difficulty score $\in [0.1\overline{6}, 0.\overline{3})$
- *hard*: Difficulty score $\in [0.\overline{3}, 0.5)$
- *very hard*: Difficulty score $\in [0.5, 1)$

Although we doubt that experiments derived from "very hard" candidate anomalies will resemble any real application domain, we decided to include them in our tests to see what impact they have on the results.

3.4 Semantic Variation and Clusteredness

Given a set of candidate "anomalous" data points, we applied the following algorithms to generate sets (of desired size) that are either widely dispersed or tightly clustered (as measured by Euclidean distance). To generate K dispersed points, we apply a facility location algorithm [9] to choose K points as the locations of the facilities. To generate K tightly clustered points, we choose a seed point at random and then compute the $K - 1$ data points that are closest to it in Euclidean distance. Note that when the point difficulty is constrained, then only candidate points of the specified difficulty are considered in this process. To quantify the clusteredness of the selected points, we measure the normalized *clusteredness*, which is defined as ratio of the sample variance of the "nominal" points to the sample variance of K selected "anomalous" points. When clusteredness is less than 1, the "anomalous" points exhibit greater semantic variance than the "normal" points. When clusteredness is greater than 1, the "anomalous" points are more tightly packed than the "normal" points (on average).

For purposes of analysis, we grouped the clusteredness scores into six qualitative levels: *high scatter* $(0, 0.25)$, *medium scatter* $[0.25, 0.5)$, *low scatter* $[0.5, 1)$, *low clusteredness* $[1, 2)$, *medium clusteredness* $[2, 4)$, and *high clusteredness* $[4, \infty)$.

3.5 Generating Benchmark Data Sets

To generate a specific data set, we choose a level of difficulty (easy, medium, hard, very hard), a relative frequency (0.001, 0.005, 0.01, 0.05, and 0.1), and a semantic variation setting (low or high). Then we apply the corresponding semantic variation procedure (with K set to achieve the desired relative frequency) to the set of available points of the desired difficulty level. For each combination of levels, we attempted to create 40 replicate data sets. However, when the number of candidate anomalous data points (at the desired difficulty level) is small, we limit the number of data sets to ensure that the replicates are sufficiently distinct. Specifically, let N be the number of available points. We create no more than $\lfloor N/K \rfloor$ replicates.

In total, from the 19 "mother" sets listed earlier, this methodology produced 4,369 problem set replicates, all of which we employed to test several statistical outlier detection algorithms.

4. ALGORITHMS

To simultaneously assess the effectiveness of our methodology and compare the performance of various statistical anomaly detection algorithms, we conducted an experimental study using several well-known anomaly detection algorithms. In this section, we describe each of those algorithms. For algorithms that required parameter tuning, we employed cross-validation (where possible) to find parameter values to maximize an appropriate figure of merit (as described below). In all cases, we made a good faith effort to maximize the performance of all of the methods. Some parameterization choices had to be made to ensure that the given algorithm implementation would return real-valued results.

4.1 One-Class SVM (ocsvm)

The One-Class SVM algorithm (Scholkopf et al. [24]) shifts the data away from the origin and then searches for a kernel-space decision boundary that separates fraction $1 - \delta$

of the data from the origin. We employ the implementation of Chang and Lin [6] available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`. For each benchmark, we employ a radial basis kernel and search parameter space until approximately 5% ($\delta = 0.05$) of the data lies outside the decision boundary in cross-validation. We would have preferred to use smaller values for δ, but OCSVM would not execute reliably for smaller values. The distance of a point from the decision boundary determines the anomaly score of that point.

4.2 Support Vector Data Description (svdd)

As proposed by Tax and Duin [25], Support Vector Data Description finds the smallest hypersphere (in kernel space) that encloses $1 - \delta$ of the data. We employed the `libsvm` implementation available at `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/` with a Gaussian radial basis function kernel. We search for parameters such that approximately 1% ($\delta = 0.01$) of the data lie outside the decision surface in cross validation. The distance of a point from the decision surface determines the anomaly score of that point.

4.3 Local Outlier Factor (lof)

The well-known Local Outlier Factor algorithm (Breunig, et al. [4]) computes the outlier score of a point x by computing its average distance to its k nearest neighbors. It normalizes this distance by computing the average distance of each of those neighbors to *their* k nearest neighbors. So, roughly speaking, a point is declared to be anomalous if it is significantly farther from its neighbors than they are from each other. We employed the `R` package `rlof` available at `http://cran.open-source-solution.org/web/packages/Rlof/`.

We chose k to be 3% of the data set. This was the smallest value for which LOF would reliably run on all data sets.

4.4 Isolation Forest (if) and Split-selection Criterion Isolation Forest (scif)

The Isolation Forest algorithm (Liu, et al. [18]) creates a forest of random axis-parallel projection trees. It derives a score based on the observation that points that become isolated closer to the root of a tree are easier to separate from the rest of the data and therefore are more likely to be anomalous. This method has a known weakness when the anomalous points are tightly clustered. To address this weakness, Liu, et al. [19] developed the Sparse-selection Criterion Isolation Forest. SCiForest subsamples the data points and features when growing each tree. An implementation was obtained from `http://sourceforge.net/projects/iforest/`.

Isolation Forest is parameter-free. For SCiForest, we chose the number of data points to subsample to be 0.66 of available data points and the number of features to consider to be 0.66 of the available features.

4.5 Ensemble Gaussian Mixture Model (egmm)

A classic approach to anomaly detection is to fit a probabilistic model to the available data to estimate the density $P(x)$ of each data point x. Data points of low density are declared to be anomalies. One approach to density estimation is to fit a Gaussian mixture model (GMM) using the EM algorithm. However, a single GMM is not very robust, and it requires specifying the number of Gaussians k. To improve robustness, we generate a diverse set of models by varying the number of clusters k, the EM initializations, and training on 15 bootstrap replicates of the data [29]. We choose a set of possible values for k, $\{6, 7, 8, 9, 10\}$, and try all values in this set. The average out-of-bag log likelihood for each value of k is computed, and values of k whose average is less than 85% of the best observed value are discarded. Finally, each data point x is ranked according to the average log likelihood assigned by the remaining GMMs (equivalent to the geometric mean of the fitted probability densities).

5. SUMMARY OF RESULTS

To assess performance, we employed the AUC (area under the ROC curve). Table 1 provides an overall summary of the algorithms. It shows the number of data sets in which each algorithm appeared in the top 3 algorithms when ranked by AUC (averaged over all settings of difficulty, relative frequency, and clusteredness). We see that Isolation Forest (IF) is the top performer, followed by EGMM and SCIF.

Table 1: # Times in Top 3

egmm	if	lof	ocsvm	scif	svdd
14	**17**	6	5	13	2

To quantify the impact of each of the design properties (relative frequency, point difficulty, and clusteredness) as well as the relative effect of each algorithm and "mother" data set, we performed an ordinary linear regression to model the *logit(AUC)* of each replicate data set as a linear function of each of these factors. The logit transform ($\log[AUC/(1 - AUC)]$) transforms the AUC (which can be viewed as a probability) onto the real-valued log-odds scale. We employed the following R formula:

$$logit(AUC) \sim set + algo + diff + rfreq + cluster \quad (1)$$

where, *set* is the dataset (*abalone*, *shuttle*, etc.), *algo* is the algorithm, *diff* is the point difficulty level, *rfreq* is the relative frequency of anomalies in the benchmark, and *cluster* is the clusteredness of the anomaly class. The *diff*, *rfreq*, and *cluster* values were binned into qualitative factors as described above. Despite the simplicity of this model, inspection of the residuals showed that it gives a reasonable fit.

We found all factors included in the regression to be significant ($p \ll 0.001$, t-test). Figure 1 shows that as the point difficulty increases, the performance degrades for all algorithms. Error bars in this and all subsequent figures show $\pm$ one standard error for the estimates from the regression. Figure 2 shows that anomalies are harder to detect as they become more frequent. And Figure 3 shows that they become harder to detect as they become more clustered. These results all confirm that the benchmark data sets achieve our design goals.

Figure 4 shows the performance on all datasets relative to *abalone*. Anomalies were hardest to detect for *yearp* and easiest for *wave*. Finally, Figure 5 shows the contribution of each algorithm to the logit(AUC) relative to EGMM. This suggests that EGMM and Isolation Forest are giving very similar performance, while the other algorithms are substantially worse.

We also fit a version of Equation (1) with pairwise interaction terms between algorithm, point difficulty, relative frequency, and clusteredness. Very few of these interactions

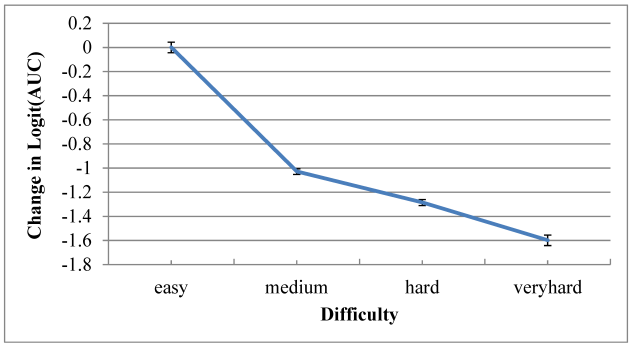


Figure 1: Change in Logit(AUC) with Difficulty

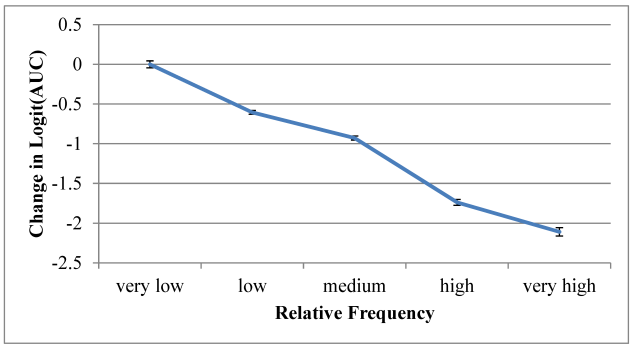


Figure 2: Change in Logit(AUC) with Rel. Freq.

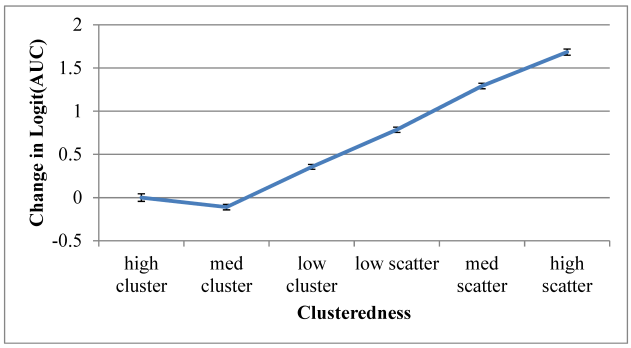


Figure 3: Change in Logit(AUC) with Clusteredness

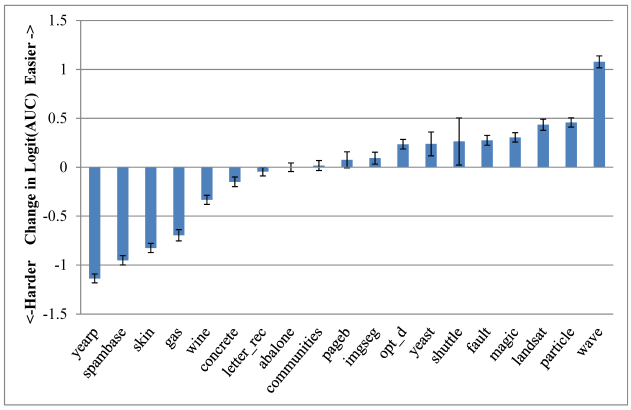


Figure 4: Performance on Datasets

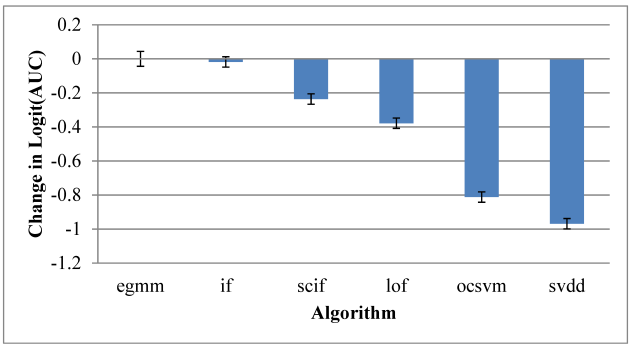


Figure 5: Change in Logit(AUC) with Algorithm

were statistically significant, which confirms that our simple model gives a good characterization of the benchmarks.

6. CONCLUSIONS

We have described a methodology for creating anomaly detection benchmarks and techniques for controlling three important properties of those benchmarks (*point difficulty*, *relative frequency* and *clusteredness*). Experimental tests based on thousands of replicate data sets demonstrate that these three properties strongly influence the behavior of several leading anomaly detection algorithms.

7. FUTURE WORK

We consider these results a work in progress and intend to develop this study further. Our plan is to include more algorithms and metrics and to provide additional statistical analysis of the results. This will include more rigorous statistical justification for our findings, an empirical comparison of algorithms, and an exploration of which settings cause shifts in the relative performance of the algorithms.

An important goal for future work is to validate the predictive value of our benchmarks against real anomaly detection problems. In particular, if we measure the point difficulty, relative frequency, and clusteredness of a real problem, does the most similar benchmark problem predict which anomaly detection algorithms will work best on the real problem? Another important goal is to develop a method for controlling the proportion of relevant (versus irrelevant) features. This would help the research community develop better methods for feature selection in anomaly detection algorithms.

8. ACKNOWLEDGMENTS

Funding was provided by the U.S. Army Research Office (ARO) and Defense Advanced Research Projects Agency (DARPA) under Contract Number W911NF-11-C-0088. The content of the information in this document does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

9. REFERENCES

[1] A. Alzghoul and M. Löfstrand. Increasing availability of industrial systems through data stream mining.

Computers & Industrial Engineering, 60(2):195 – 205, 2011.

[2] K. Bache and M. Lichman. UCI machine learning repository, 2013.

[3] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[4] M. Breunig, H.-P. Kriegel, R. T. Raymond T. Ng, and J. Sander. LOF: identifying density-based local outliers. *ACM SIGMOD Record*, pages 93–104, 2000.

[5] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 41(3):15:1–15:58, July 2009.

[6] 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.

[7] E. Dereszynski and T. G. Dietterich. Spatiotemporal models for anomaly detection in dynamic environmental monitoring campaigns. *ACM Transactions on Sensor Networks*, 8(1):3:1–3:26, 2011.

[8] J. Glasser and B. Lindauer. Bridging the gap: A pragmatic approach to generating insider threat data. In *2013 IEEE Security and Privacy Workshops*, pages 98–104. IEEE Press, 2013.

[9] T. F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38(0):293 – 306, 1985.

[10] J. Greensmith, J. Twycross, and U. Aickelin. Dendritic cells for anomaly detection. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 664–671. IEEE, 2006.

[11] V. J. Hodge and J. I. M. Austin. A survey of outlier detection methodologies. *AI Review*, 22:85–126, 2004.

[12] T. Jaakkola and D. Haussler. Probabilistic kernel regression models. In *Proceedings of the 1999 Conference on AI and Statistics*, volume 126, pages 00–04. San Mateo, CA, 1999.

[13] S. Keerthi, K. Duan, S. Shevade, and A. Poo. A fast dual algorithm for kernel logistic regression. *Machine Learning*, 61(1-3):151–165, 2005.

[14] J. S. Kim and C. Scott. Robust kernel density estimation. In *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pages 3381–3384, 2008.

[15] T. Lane and C. E. Brodley. Sequence matching and learning in anomaly detection for computer security. In *AAAI Workshop: AI Approaches to Fraud Detection and Risk Management*, pages 43–49, 1997.

[16] A. Lazarevic, L. Ertoz, V. Kumar, A. Ozgur, and J. Srivastava. A comparative study of anomaly detection schemes in network intrusion detection. In *In Proceedings of SIAM Conference on Data Mining*, 2003.

[17] A. Liu, C. Martin, T. Hetherington, and S. Matzner. A comparison of system call feature representations for insider threat detection. In *Information Assurance Workshop, 2005. IAW '05. Proceedings from the Sixth Annual IEEE SMC*, pages 340–347, 2005.

[18] F. T. Liu, K. M. Ting, and Z.-H. Zhou. Isolation forest. In *Proceedings of the IEEE International Conference on Data Mining*, pages 413–422, 2008.

[19] F. T. Liu, K. M. Ting, and Z.-H. Zhou. On detecting clustered anomalies using SCiForest. In *Machine Learning and Knowledge Discovery in Databases*, pages 274–290, 2010.

[20] M. Markou and S. Singh. Novelty detection: a review - part 1: statistical approaches. *Signal Processing*, 83(12):2481–2497, 2003.

[21] D. Pokrajac, A. Lazarevic, and L. Latecki. Incremental local outlier detection for data streams. In *Computational Intelligence and Data Mining, 2007. CIDM 2007. IEEE Symposium on*, pages 504–515, 2007.

[22] K. Polat, S. Sahan, H. Kodaz, and S. Günes. A new classification method for breast cancer diagnosis: Feature selection artificial immune recognition system (fs-airs). In L. Wang, K. Chen, and Y. Ong, editors, *Advances in Natural Computation*, volume 3611 of *Lecture Notes in Computer Science*, pages 830–838. Springer Berlin Heidelberg, 2005.

[23] L. Portnoy, E. Eskin, and S. Stolfo. Intrusion detection with unlabeled data using clustering. In *In Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001*. Citeseer, 2001.

[24] B. Schölkopf, J. C. Platt, J. Shawe-taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution, 1999.

[25] Tax and Duin. Support vector data description. *Machine Learning*, 54:45–66, 2004.

[26] K. L. Wagstaff, N. L. Lanza, D. R. Thompson, T. G. Dietterich, and M. S. Gilmore. Guiding scientific discovery with explanations using DEMUD. In *Proceedings of the Association for the Advancement of Artificial Intelligence AAAI 2013 Conference*, 2013.

[27] F. Xue, W. Yan, N. Roddy, and A. Varma. Operational data based anomaly detection for locomotive diagnostics. In *International Conference on Machine Learning*, pages 236–241, 2006.

[28] B. Zhang, C. Sconyers, C. Byington, R. Patrick, M. Orchard, and G. Vachtsevanos. Anomaly detection: A robust approach to detection of unanticipated faults. In *Prognostics and Health Management, 2008. PHM 2008. International Conference on*, pages 1–8, 2008.

[29] Z.-H. Zhou. *Ensemble Methods: Foundations and Algorithms*. Chapman and Hall/CRC, 2012.

[30] J. Zhu and T. Hastie. Kernel logistic regression and the import vector machine. In *Journal of Computational and Graphical Statistics*, pages 1081–1088. MIT Press, 2001.

[31] A. Zimek, E. Schubert, and H.-P. Kriegel. A survey on unsupervised outlier detection in high-dimensional numerical data. *Statistical Analysis and Data Mining*, 5(5):363–387, 2012.

Anomaly Detection on ITS Data via View Association

Junaidillah Fadlil
nedijf@gmail.com

Hsing-Kuo Pao
pao@mail.ntust.edu.tw

Yuh-Jye Lee
yuh-jye@mail.ntust.edu.tw

National Taiwan University of Science and Technology
No. 43, Sec. 4, Keelung Rd.
Taipei, Taiwan 106

ABSTRACT

We focus on detecting anomalous events in transportation systems. In transportation systems, other than normal road situation, anomalous events happen once in a while such as traffic accidents, ambulance car passing, harsh weather conditions, etc. Identifying the anomalous traffic events is essential because the events can lead to critical conditions where immediate investigation and recovery may be necessary. We propose an anomaly detection method for transportation systems where we create a police report automatically after detecting anomalies. Unlike the traditional police report, in this case, some quantitative analysis shall be done as well to provide experts with an advanced, precise and professional description of the anomalous event. For instance, we can provide the moment, the location as well as how severe the accident occurs in the upstream and downstream routes. We present an anomaly detection approach based on view association given multiple feature views on the transportation data if the views are more or less independent from each other. For each single view, anomalies are detected based on a manifold learning and hierarchical clustering procedures and anomalies from different views are associated and detected as anomalies with high confidence. We study two well-known ITS datasets which include the data from Mobile Century project and the PeMS dataset, and we evaluate the proposed method by comparing the automatically generated report and real report from police during the related period.

Categories and Subject Descriptors

I.5.2 [**Pattern Recognition**]: Design Methodology—*Pattern analysis*; H.4.2 [**Information Systems Applications**]: Types of Systems—*Decision support (e.g., MIS)*

General Terms

Algorithms

ODD'13, August 11th, 2013, Chicago, IL, USA.

Keywords

Anomaly Detection, Intelligent Transportation System (ITS), Association, Trajectory, Manifold learning.

1. INTRODUCTION

We have entered an era where sensor devices are massively utilized to monitor the environment around us. Sensors can communicate between each other and can communicate with backend systems as well. Based on that, we can distributively collect data from regional sensor readings to profile regional patterns for further follow-up examinations. In transportation monitoring, not too long ago people in USA call 911 to report accidents in traffic when accidents occur. As time goes by, deployed sensors on roadside are now commonly used for traffic information collection. Given the traffic information, we can understand traffic status so that traffic police and drivers like us can take appropriate actions afterwards, so called the Intelligent Transportation System (ITS). In recent applications, smartphone devices have also been included as part of the monitoring system.

Since roads are covered by the sensor-based information system, many technologies have been applied. Examples include incident detection systems [13] where detection of incident can significantly reduce the number of unnecessary highway patrols; automatic plate number recognition [3] for the purpose of surveillance and traveling time estimation; traffic signal control system [14] to help us for traffic flow optimization[1]. Many of the above technology can also be combined together as an integrated system to built a more complex ITS. In metropolitan area, such ITS becomes necessary in all respects. In this work, we focus on incident detection based on an anomaly detection approach.

Given the data collected on an ITS, the purpose of this study is to detect anomalies within the traffic which may be due to incidents, and to estimate the influence of anomalous events in nearby incident area such as upstream and downstream routes of the incident location. That is, we intent to produce a report automatically that is similar to a police report which includes all necessary information about an incidents or an anomalous event; furthermore, we would like to add additional quantitative information to extend the police report. For instance, a police report may record information merely about an accident including the accident location, when the accident happened, and what kind of accident such as traffic collision with unknown reason, hit and

[1]Zhang et al. [22] reported that almost 40% of the population spends at least one hour on the road each day in USA.

run, traffic collision with injuries, and so on. However, the accident might affect the nearby area, as illustrated in Figure 1. As depicted in the illustration, traffic in upstream routes is likely to be more severely congested than that in the downstream routes. This kind of information, even it is useful for drivers and police, is usually not included in the police or ITS report.

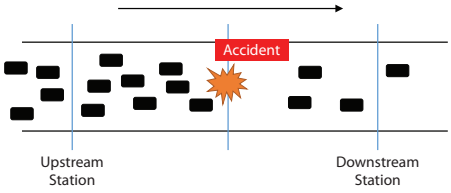


Figure 1: Illustration of traffic accident, and the status near the upstream and downstream routes.

We propose an anomaly detection method that can detect traffic anomalies by feature view association. Given a multi-viewed dataset, we assume that the information of different views are collected separately and there exists no *contextual anomalies* across different views[2] and the anomalies can be found within each single view. Based on a few anomaly detection results from different views, and to associate anomalies detected from those views, we can confirm the anomalies with high confidence.

The evaluation is done mainly on the *Mobile Century* dataset which is a well-known ITS dataset for traffic analysis. Moreover, to apply the proposed method to a relatively large-scale dataset, and to further study the spatial and temporal relationship between data, we also test the proposed method on the *PeMS* data, another well-known ITS dataset. Details of the two datasets are shown in Section 3.1.

The benefit of the proposed anomaly detector can be summarized as follows:

1. Different from most of the one-class anomaly detection methods, the proposed method needs very few parameters or threshold tuning to decide how likely to be considered as normal patterns.
2. In principle, the proposed method needs no "clean" data for the training of normal patterns. In general, the search of clean data can be difficult, or as arbitrary as suggested by subjective domain experts.
3. The computation of the proposed method is efficient in the sense that the computation on each single feature view can be done separately. Therefore, we can easily extend the algorithm to a parallel version when multiple-process or parallel computation is available.

We expect that the proposed method can be applied to applications other than the traffic incident detection. We should also expect that the proposed method can be extended to solve the anomaly detection task on large-scale datasets.

Before we go on to introduce the proposed method, we discuss some previous works on anomaly detection and traffic analysis in Section 2; after that, we present the datasets that we use in this work in Section 3.1, and in Section 3.2, we describe the proposed method. The experiment result is shown in Section 4 and in Section 5, we conclude our presentation.

[2]There may exist contextual anomalies *within* a single feature view though.

2. RELATED WORK

Many studies have focused on traffic analysis and one major approach is to detect anomalous events from traffic patterns. For example, given taxi trajectories recorded from GPS, Chawla et al. [4] proposed a framework to infer the main reason why some anomalies appear in road traffic data. In their framework, they modeled the road structures as a directed graph then employed PCA algorithm to detect anomalies. Chen et al. [5] proposed *iBOAT* that can detect anomalous trajectories "on-the-fly". They extracted useful information from the behaviors of urban road users, and analyzed adverse or possibly malicious events such as a driver taking a questionable route.

In many anomaly detection schemes, an effective data representation can reveal the difference between normal and anomalous patterns. Thajchayapong et al. [20] monitored traffic anomalies using microscopic traffic variables such as *relative speed* and *inter-vehicle spacing*. By using Gaussian process to model the microscopic traffic variables, they can grab temporary changes in the traffic pattern and detect anomalies.

Several techniques have been proposed to detect anomalies in video surveillance. Fu et al. [8] proposed using a hierarchical clustering framework to classify vehicle motion trajectories in real traffic video. They showed that their proposed method performs better than the conventional fuzzy K-means clustering. Piciarelli et al. [15] clustered the trajectories in an online fashion, and modeled the trajectory data in a tree-like structure where some probability information is also included. Jiang et al. [10] proposed video event detection based on unsupervised clustering of object trajectories, which are modeled by Hidden Markov Model [16]. This study employ a dynamic hierarchical process for trajectories clustering to prevent model overfitting together with a 2-depth greedy search for efficient clustering.

Similar to our approach, Agovic et al. [1] investigated an anomalous cargo using a manifold embedding method for feature representation. Although, they focused on both linear and nonlinear methods, the paper results show that nonlinear methods outperform the linear methods. Also related to our research, Kind et al. [12] proposed feature-based anomaly detection that constructs histograms of different traffic features. In a survey paper for the outdoor surveillance task, Zhang et al. [23] compared six different similarity measures that are used for trajectory clustering. They showed that the hybrid PCA [11] and Euclidean distance combined method outperforms other methods; and PCA method is very sensitive to its parameters. Ringberg et al. [17] studied the sensitivity of PCA for the anomaly detection. They pointed several challenges in their work such as evaluating how sensitive the false positive rate to small differences of the dimensionality in normal space, and to the level of aggregation in traffic measurement. Danilo et al. [21] studied the estimation of cellular network to build road traffic system. Hence, based on the explorative analysis of real-time signaling data the result showed whether the traffic is normal or abnormal.

To speak of methodology, the proposed method is also inspired by the well-known *co-training* algorithm developed by Blum and Mitchell [2] for semi-supervised learning. The

co-training algorithm splits data attributes into several subsets, given the assumption that the attribute subsets are conditionally independent with the known label information. Each subset plays a view and is *sufficient* to learn a classifier; therefore, it can use the prediction from one view to help other views to learn the label information of unlabeled data. The multi-view approach proposed in this work is similar to the co-training method in the sense that we also use information from different views to decide anomalies.

3. DATASETS AND PROPOSED METHOD

In this section, we explain the proposed anomaly detection method in detail. In order to build intuition on the method, we describe the datasets that are used in this study before the method and then we can illustrate ideas through concrete examples.

3.1 Datasets

We evaluate the proposed method through two datasets: the first dataset is the one from *Mobile Century* project [9], a traffic dataset that was collected on February 8, 2008 along a 10-mile stretch of I-880 highway near Union City, California, USA, for over eight hours (10:00 AM - 18:00 PM). In this work, we focus on two parts of the dataset, the *GPS* individual trajectories and the loop detector *PeMS* data. The second dataset is the *PeMS* dataset from California Department of Transportation (Caltrans) website[3]. The Caltrans website has been keeping records starting from 1993. In order to differentiate between *PeMS* data from *Mobile Century* project and *PeMS* data from California DOT, we call the first *PeMS Century PeMS* and and the latter *PeMS* data *Caltrans PeMS*.

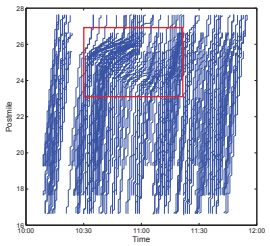


Figure 2: Plot of individual trajectories produced by *GPS*; the red square indicates the space and time information of an accident. In the *Caltrans PeMS* website, the accident report recorded the accident for only one station (postmile 26.641); however, as shown in this plot, the accident propagates and gives effect to the upstream and (a little to) the downstream stations.

Figure 2 shows the *GPS* data that indicates where and when the accident happened. The *Century PeMS* data were recorded for every 30 seconds while *GPS* data were recorded for every 3 seconds. The *GPS* data contains information about latitude and longitude. We shall use the information to associate with the location information of *PeMS* stations. Moreover, we just sample the trajectories that start from 10:00 AM and end at 11:45 AM as our data in this study.

[3]http://pems.dot.ca.gov/

The data provided by the *Caltrans PeMS* website are aggregated for every 5 minutes. In the *Caltrans PeMS* experiment, we select data close to postmile 26.641 (station 400165), starting from 10:00 AM until 18:00 PM.

Ground Truth.

The ground truth was obtained from *Caltrans PeMS* website which records incidents on several California freeways including accidents, traffic hazards, congestions, traffic breakdowns, and so on. On the *Mobile Century* data for the first experiment, we have an accident reported on postmile 26.641, occurred from 10:34 AM, February 8, 2008, with a duration of 34 minutes. In the second experiment, the *Caltrans PeMS* reported an accident on the same postmile (26.641), which occurred at 1:00 PM, December 14, 2007, with a duration of 38 minutes.

3.2 Method

In this subsection, we explain the proposed anomaly detection method in full details. As shown in Figure 3, the main purpose of our work is to create a report for traffic incidents given different *views* (features) of data. The proposed method consists of four steps. The first step is to extract useful features from the data. Second, we utilize a manifold embedding method called Isomap [19] for data representation. After that, in the third step, we cluster the projected points using a hierarchical clustering method [18]. We detect anomalies based on the hierarchical clustering result and that is done for each single feature view. Overall, we may have anomalies that are detected based on several individual views. In the end, the last step is to automatically create a report based on the different views' hierarchical clustering and anomaly detection result obtained from the previous step. To produce the final report, we associate anomalies that are detected from different views and make the final call of anomalies if they belong to the anomalous group in many different views. By detecting anomalies from different views, we believe that we can have high confidence on making the final decision which may include dispatching a police patrol to the accident location for further investigation and accident recovery.

3.2.1 Feature Extraction

In this study we use three views for anomaly detection to create incident report in *Mobile Century* data. The first view is based on the *flow* information and the second view is based on the *speed* information, which are obtained from *Century PeMS* and *GPS* data respectively. The third view is based on the *duration* information, derived from *GPS* data. We use two views for anomaly detection in *Caltrans PeMS* data. The first view is *flow* and the second view is *speed*. Feature extraction on each view is defined as follows:

- **Flow.** The *flow* data are obtained by temporal sensors. We use an appropriate time window size w to extract the features. Let $Q = \{x_1, \ldots, x_T\}$ to be a T-length time series of *flow*, we discuss many derived

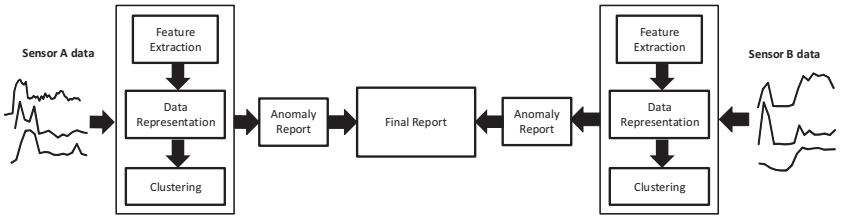


Figure 3: The proposed anomaly detection method. The final report is created based on the anomalies detected from different views, such as sensor readings of different locations, different types of sensors, different measurements, etc.

features as follows.

$$Q = \bigcup_{i=1}^{N} q_i$$

q_i : $\{x_1^i, x_2^i, \ldots, x_{k+1}^i, \ldots, x_{|q_i|}^i\} = \{x_j, x_{j+1}, \ldots\}$ for some j, data in i-th window

$|q_i|$: the number of data in q_i

N : the number of windows

w : size of each window, $\frac{T}{w} = N$.

Moreover, we define mean of q_i as:

$$m_{flow}^i = \frac{\sum_{x_j \in q_i} x_j}{|q_i|}, \quad (1)$$

also, standard deviation of q_i as:

$$s_{flow}^i = \sqrt{\frac{1}{|q_i|-1} \sum_{x_j \in q_i}^{|q_i|} (x_j - m_{flow}^i)^2}, \quad (2)$$

and skewness of q_i as:

$$g_{flow}^i = \frac{\sum_{x_j \in q_i}^{|q_i|} (x_j - m_{flow}^i)^3}{(|q_i|-1){s_i}^3}. \quad (3)$$

On the other hand, we also compute the difference between the (j-1)-th and the j-th *flow* values for each window q_i. We define L_{flow}^i as:

$$L_{flow}^i = (\ell_1^i, \ldots, \ell_{|q_i|-1}^i) \quad (4)$$

where $\ell_k^i = x_{k+1}^i - x_k^i, k = 1, \ldots, |q_i| - 1$. The features extracted from L_{flow}^i is mean $m_{\triangle flow}^i$, standard deviation $s_{\triangle flow}^i$ and skewness $g_{\triangle flow}^i$. Note that the $|q_i|$ will be the same for each window.

- **Speed.** The *speed* data are obtained from spatio-temporal sensors. First we associate the locations of *GPS* data with the *PeMS* station locations. After that, we collect another set of features that are related to *speed* information. We collect a *speed* time series $V = (v_1, \ldots, v_K)$ with length K. From the data associated with each station, we extract six features:

$$V = \bigcup_{i=1}^{M} p_i$$

p_i : set of speed data in station i

$|p_i|$: number of data in station i

M : number of stations

Table 1: Summary of feature extraction

View	Data Source	Feature
Flow	*Century PeMS*	1. mean of flow 2. std. of flow 3. skewness of flow 4. mean of △flow 5. std. of △flow 6. skewness of of △flow
Speed	*Century GPS* (trajectory)	1. mean of speed 2. std. of speed 3. skewness of speed 4. mean of △speed 5. std. of △speed 6. skewness of of △speed
Duration	*Century GPS* (trajectory)	1. mean of duration 2. std. of duration 3. skewness of duration 4. total duration
Flow	*Caltrans PeMS*	similar to *Century PeMS*
Speed	*Caltrans PeMS*	similar to *Century GPS*

the mean of p_i as m_{speed}^i, the standard deviation of p_i as s_{speed}^i and the skewness of p_i as g_{speed}^i. Similar to the *flow* view we also compute mean, standard deviation and skewness of △speed between the (i-1)-th and the i-th *speed* data. Note that data size for each Station can be different, unlike the case when we collect *flow* features where we have identical number of data in each window.

- **Duration**. In our study we compute duration between the (i-1)-th data and the i-th data for each station that has been passed by vehicle with GPS-enabled smartphones. The way to extract the feature is similar to the *speed* view, but we only extract four features: mean, standard deviation, skewness and total duration to pass a station.

Table 1 gives a summary of the complete feature set. Since we have the data from different features, we normalize them into the range of $[0, 1]$.

3.2.2 Data Representation

Given the features computed in previous subsubsection, first, we compute Euclidean distances between each pair of data points. After that, we utilize a manifold learning method called Isomap for data representation. The Isomap method consists of three steps:

- Construct a neighborhood graph based on k-nearest neighbor (kNN) information.
- Create the shortest path between each pair of data

points. This can be done by, for instance, Dijkstra's shortest path algorithm.
- Apply Multidimensional Scaling (MDS) [6] to find low-dimensional embeddings for data points.

In our study data representation plays an important role. We assume that the space, so-called the intrinsic space is better than the original space to show the relationship between data points. Some detection techniques are indeed more effective if worked on low-dimensional intrinsic space [7]. Moreover, a low-dimensional representation is often desirable for experts to visualize data relationship. To speak of the efficiency issue, working on low-dimensional space usually has low computing complexity.

3.2.3 Data Clustering

Given the projected data in low-dimensional space, we use hierarchical clustering method to cluster the projected data into groups and split the data into two clusters. In our opinion, the normal and anomalous patterns should show difference on the clustering result and therefore be separated into different clusters; hopefully, one for the normal cluster and the other for the anomalous cluster. In real life, the number of anomalous events is usually much smaller than the number of normal events. Hence, to further improve the clustering result we can apply the so-called 90-10 rule. This rule means the normal group should include at least 90% of the whole data points and the anomalous group may include only at most 10% of the whole set. We can confirm the clustering result by the rule to guarantee it is a detection of anomalous behavior rather than a classification of data points into two or more types. Figure 4(b), 5(b) and 6(b) illustrate how clustering has been done and they also show the anomalous events have fewer points compared to the normal points.

3.2.4 Final Anomaly Report

The data will be labeled based on the result of data clustering, for each single view. Afterward, associating the results from two or more views by computing their intersection, will generate final result automatically. That means if all views detect some anomalies occurred in the same location and at the same time, it will be considered as the final predicted anomalies. The final anomaly report contains information about time and location when and where the anomalies happened; also some influence of the anomalies will be included as well.

4. EXPERIMENTS

We divide the experiments into two parts. First, we use the *Mobile Century data* to evaluate the proposed method. We utilize four combined views to detect an anomalous event that happened on Feb. 8, 2008, and the combined views include: ($flow$, $speed$), ($speed$, $duration$), ($flow$, $duration$), and ($flow$, $speed$, $duration$). Second, we study the performance of the proposed method on the *Caltrans PeMS* dataset. In this study we discuss the performance of the proposed method on detecting an accident that happened on Dec. 14, 2007 and we use the only applicable views, namely $flow$ and $speed$ views for the detection. In addition, we show some preliminary online learning result that use previous days to train a model and then test on a later day when the accident happened.

4.1 Experimental Settings

In this series of experiments we set the number of neighbors to be five for kNN which is used in Isomap to build the neighborhood graph, and we set the window size as five minutes and 30 minutes for the *Mobile Century* data and the *Caltrans PeMS* data respectively. In data clustering we use Euclidean distance to compute pairwise distance between points, and use the shortest distance as the distance between clusters to create a cluster tree. Table 2 shows a summary of our experimental settings.

4.2 Experiment Results

4.2.1 Evaluation via the Mobile Century Data

In this section we show the evaluation result given the *Mobile Century* data. We show how we detect anomalies given different view combinations. Some low-dimensional data representation shows that the normal and anomalous patterns are well separated from each other and the proposed method is effective for the detection. Table 2 shows the *PeMS* stations of interest, where the traffic near the stations may be affected by the accident. Note that the dataset is a one-day dataset.

Table 3 shows all the anomaly detection results from each of the following views: (a) *flow* from *Century PeMS*, (b) *speed* from *Century GPS*, and (c) *duration* from *Century GPS*, on six stations. The results indicate that the anomalous events can be detected based on each single view, just may not be perfect, still, we can observe how an accident affects the traffic along the road. As shown previously in Figures 1 and 2 that both of the upstream and downstream traffic can be affected by accidents if there is any. In Figure 2, the red square indicates the propagation of traffic flow from the accident. The y-axis indicates that the traffic close to the nearby postmiles or locations may also become affected. We can observe this kind of information in our detection results; however, this information is usually not recorded in the police patrol report which we consider as ground truth in this study. We can further improve the results if more than one view are considered simultaneously.

Table 4 shows view association results[4] based on different combined views: (a) *flow* and *speed*, (b) *flow* and *duration*,(c) *speed* and *duration*, and (d) *flow*, *speed* and *duration* all together. Overall, the result of anomaly detection by associating *flow* and *speed* views gives the best result if compared to that of all other combinations. On station 400165 where the accident happened, the result from *flow* and *speed* view association reports that the anomalous event starts from 10:35 AM and ends at 10:50 AM, which is very close to the ground truth where it indicates an accident from 10:34 AM to 11:08 AM. In addition, the table shows that the results from all view associations share high correlation to each other.

We also observe that the anomalous and normal patterns are indeed different if represented in low-dimensional space. In Figures 4, 5, and 6 the results are from *flow*, *speed* and *duration* view respectively. The figures show that the anomalous patterns are well separated from the normal ones. The labeling information is just used for visualization and not

[4]Note that the results produced by the *GPS* data and *PeMS* may have different resolutions. Hence, we have to round the data into five minute-based one before computing their association.

Table 2: Summary of experimental settings: $k_{Iso} = 5$ for kNN in Isomap, and the intrinsic dimensionality is set to 2. Data are collected for each 5 or 30 mins to compute the mean, standard deviation, etc., on each station, for *Mobile Century* data or *Caltrans* data respectively.

	Dataset	View	Data Interval	Station ID (Postmile)
Exp. 1	*Century PeMS*	flow	5 mins, 1 station	400488 (24.007) 401561 (24.477) 400611 (24.917) 400284 (25.767) 400041 (26.027) 400165 (26.641)
	Century GPS	speed	1 station	
	Century GPS	duration	1 station	
Exp. 2	*Caltrans PeMS*	flow, speed	30 mins, 1 station	400165 (26.641)

Table 3: The reports produced from different single views, namely *flow*, *speed* and *duration*.

400488 (24.007)	401561(24.477)	400611(24.917)	400284(25.767)	400041(26.027)	400165(26.641)
2/8/2008 10:50	2/8/2008 10:00	2/8/2008 10:00	2/8/2008 10:30	2/8/2008 10:25	2/8/2008 10:25
2/8/2008 10:55	2/8/2008 10:45	2/8/2008 10:40	2/8/2008 10:35	2/8/2008 10:30	2/8/2008 10:30
2/8/2008 11:00	2/8/2008 10:50	2/8/2008 10:45	2/8/2008 10:40	2/8/2008 10:35	2/8/2008 10:35
	2/8/2008 10:55	2/8/2008 10:50	2/8/2008 10:45	2/8/2008 10:40	2/8/2008 10:40
	2/8/2008 11:00	2/8/2008 10:55	2/8/2008 10:50	2/8/2008 10:45	2/8/2008 10:45
		2/8/2008 11:00	2/8/2008 10:55	2/8/2008 10:50	2/8/2008 10:50
				2/8/2008 10:55	

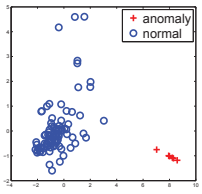
(a) The report produced by *flow* view (*Century PeMS* stations)

400488 (24.007)	401561(24.477)	400611(24.917)	400284(25.767)	400041(26.027)	400165(26.641)
2/8/2008 10:45	2/8/2008 10:40	2/8/2008 10:37	2/8/2008 10:43	2/8/2008 10:43	2/8/2008 10:38
2/8/2008 10:47	2/8/2008 10:41	2/8/2008 10:41	2/8/2008 10:46	2/8/2008 10:51	2/8/2008 10:39
2/8/2008 10:49	2/8/2008 10:42	2/8/2008 10:43	2/8/2008 10:48	2/8/2008 10:54	2/8/2008 10:44
2/8/2008 10:50	2/8/2008 10:46	2/8/2008 10:44	2/8/2008 10:53	2/8/2008 10:56	2/8/2008 10:44
2/8/2008 10:51	2/8/2008 10:49	2/8/2008 10:45	2/8/2008 10:55	2/8/2008 10:58	2/8/2008 10:50
	2/8/2008 10:51	2/8/2008 10:49	2/8/2008 10:56	2/8/2008 10:59	2/8/2008 10:55
	2/8/2008 10:53	2/8/2008 10:58	2/8/2008 10:58	2/8/2008 11:06	2/8/2008 11:00
					2/8/2008 11:01
					2/8/2008 11:07

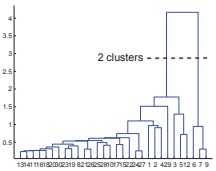
(b) The report produced by *speed* view (GPS data)

400488 (24.007)	401561(24.477)	400611(24.917)	400284(25.767)	400041(26.027)	400165(26.641)
2/8/2008 10:45	2/8/2008 10:39	2/8/2008 10:37	2/8/2008 10:29	2/8/2008 10:38	2/8/2008 10:50
2/8/2008 10:47	2/8/2008 10:40	2/8/2008 10:40	2/8/2008 10:30	2/8/2008 10:43	2/8/2008 10:55
2/8/2008 10:50	2/8/2008 10:41	2/8/2008 10:41	2/8/2008 10:33	2/8/2008 10:45	
	2/8/2008 10:42	2/8/2008 10:43	2/8/2008 10:35	2/8/2008 10:47	
	2/8/2008 10:44	2/8/2008 10:44	2/8/2008 10:40	2/8/2008 10:51	
	2/8/2008 10:46	2/8/2008 10:45	2/8/2008 10:43		
	2/8/2008 10:48	2/8/2008 10:48	2/8/2008 10:46		
	2/8/2008 10:50	2/8/2008 10:51	2/8/2008 10:50		
	2/8/2008 10:55	2/8/2008 10:58	2/8/2008 10:55		

(c) The report produced by *duration* view (GPS data)

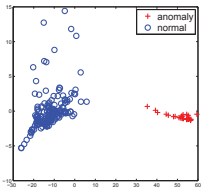


(a) *Flow* view (PeMS)

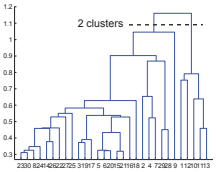


(b) Dendrogram of *flow* view

Figure 4: *Flow* view results from PeMS Station 400165 where the accident happened (a) the Isomap data plot of *flow* view (b) dendrogram of hierarchical clustering of *flow* view.



(a) *Speed* view (GPS)



(b) Dendrogram *speed* view

Figure 5: *Speed* view results from GPS phones (a) the Isomap data plots of *speed* view (b) dendrogram of hierarchical clustering of *speed* view.

used in our experiments. The anomaly detection is based on the hierarchical clustering result shown on the right-hand side of Figures 4-6. To speak of the detection on spatial domain, Figure 7 shows that the *flow* view detection result from different *PeMS* stations which were affected by accident. We adopt the 90-10 rule which was discussed in Sub-

Table 4: The reports produced by all combinations of view association.

400488 (24.007)	401561(24.477)	400611(24.917)	400284(25.767)	400041(26.027)	400165(26.641)
2/8/2008 10:50	2/8/2008 10:45	2/8/2008 10:40	2/8/2008 10:40	2/8/2008 10:40	2/8/2008 10:35
2/8/2008 10:55	2/8/2008 10:50	2/8/2008 10:45	2/8/2008 10:45	2/8/2008 10:45	2/8/2008 10:40
	2/8/2008 10:55	2/8/2008 10:50	2/8/2008 10:50	2/8/2008 10:50	2/8/2008 10:45
			2/8/2008 10:55	2/8/2008 10:55	2/8/2008 10:50

(a) The report produced by associating the *flow* and *speed* view

400488 (24.007)	401561(24.477)	400611(24.917)	400284(25.767)	400041(26.027)	400165(26.641)
2/8/2008 10:50	2/8/2008 10:45	2/8/2008 10:45	2/8/2008 10:30	2/8/2008 10:35	2/8/2008 10:50
	2/8/2008 10:50	2/8/2008 10:50	2/8/2008 10:35	2/8/2008 10:40	
	2/8/2008 10:55	2/8/2008 10:55	2/8/2008 10:40	2/8/2008 10:45	
			2/8/2008 10:45		
			2/8/2008 10:50		
			2/8/2008 10:55		

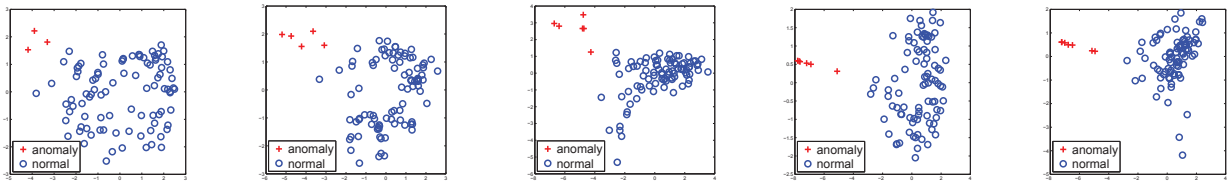
(b) Final report produced by associating the *flow* and *duration* view

400488 (24.007)	401561(24.477)	400611(24.917)	400284(25.767)	400041(26.027)	400165(26.641)
2/8/2008 10:45	2/8/2008 10:40	2/8/2008 10:37	2/8/2008 10:43	2/8/2008 10:43	2/8/2008 10:50
2/8/2008 10:47	2/8/2008 10:41	2/8/2008 10:41	2/8/2008 10:46	2/8/2008 10:51	2/8/2008 10:55
2/8/2008 10:50	2/8/2008 10:42	2/8/2008 10:43	2/8/2008 10:55		
	2/8/2008 10:46	2/8/2008 10:45			
	2/8/2008 10:50	2/8/2008 10:58			

(c) Final report produced by associating *speed* and *duration* view

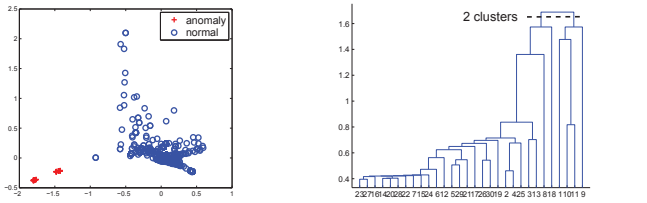
400488 (24.007)	401561(24.477)	400611(24.917)	400284(25.767)	400041(26.027)	400165(26.641)
2/8/2008 10:50	2/8/2008 10:45	2/8/2008 10:40	2/8/2008 10:45	2/8/2008 10:45	2/8/2008 10:50
	2/8/2008 10:50	2/8/2008 10:45	2/8/2008 10:50		
	2/8/2008 10:55	2/8/2008 10:50	2/8/2008 10:55		
		2/8/2008 10:55			

(d) Final report produced by associating all views: *flow*, *speed* and *duration* view



(a) Station 400488 (b) Station 401561 (c) Station 400611 (d) Station 400284 (e) Station 400041

Figure 7: The Isomap data plot for each of interested PeMS stations. The blue circle indicates normal pattern and the red cross indicates the anomalous pattern. For each station, we can observe that the anomalous points are well separated from the normal points.



(a) *Duration* view (GPS) (b) Dendrogram of *duration* view

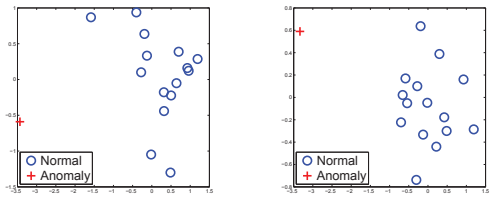
Figure 6: *Duration* view results from GPS phones (a) the Isomap data plots of *duration* view (b) dendrogram of hierarchical clustering of *duration* view.

subsection 3.2.3 to confirm the anomalies in all our detection procedures.

4.2.2 Evaluation via the Caltrans PeMS Data

In the second series of experiments, we evaluate the proposed anomaly detection scheme using the *Caltrans PeMS* data. As a preliminary study, we only focus on the data that are collected near the postmile 26.641. The proposed method detects an anomalous event at 1:00 PM, which coincides with the ground truth. Figure 8 shows the Isomap data plots for *flow* and *speed* views. Both views show that there is an unusual pattern on the left which is the accident happened at 1:00 PM of December 14, 2007. The view association confirms the finding from these two views.

Given the *Caltrans PeMS* data, a multiple-day dataset, we can test how the proposed method is used for online anomaly detection: detecting anomalous events on a later day given the training of previous days. We would like to answer the following questions: i) Given a location and a specific moment, can we detect anomalies in that moment using previous days' data for training? ii) Following the previous question, will the result be influenced if some weekend days are added in the training?



(a) *Speed* view (b) *Flow* view

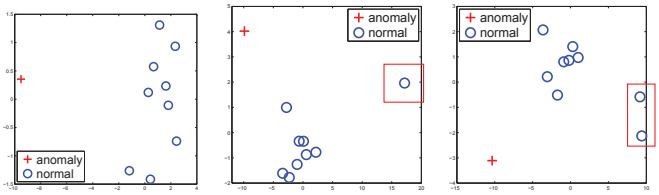
Figure 8: The anomaly detection results for Dec. 14, 2007. Both views can detect the anomaly correctly which is the accident happened at 1:00 PM. The anomalous pattern is clearly separated from other normal patterns.

Table 5: Anomaly detection results using the *speed* data on previous days.

Weekdays	One-day weekend	Two-day weekend
12/14/2007 13:00	12/9/2007 13:00	12/8/2007 13:00 12/9/2007 13:00

In order to answer the above questions, we design the experiments as follows: choosing the previous days' data for training, but (i) including only the data at the same time with the moment that we want to detect; ii) including only the data contains no accident; iii) using the previous 10 days' data; iv) detecting by hourly basis instead of daily basis. These previous days' data is used to judge if the data from this moment is considered as an anomaly or not. We use only the *speed* information in this part of study.

Table 5 shows anomaly detection using previous days' data. If we include only the weekdays for training, then we can correctly capture the anomalous event; on the other hand, if we include the weekdays' as well as the weekend's data for training, the detection result is no longer correct. The weekend's data are detected as anomalies in this case. We have more than one kind of "anomalies" and the anomaly detection procedure may detect either one of the anomalies. The data plot results are in Figure 9. Figure 9 (a) shows the detection result given only weekdays for training. Apparently, the anomaly is correctly detected in this case. Figure 9(b) and (c) show how the result is influenced by the weekend's data. The weekend's data (red square) are detected as anomalies in this case, while the real anomaly (red cross) is the one on the left.



(a) weekdays (b) one-day weekend (c) two-day weekend

Figure 9: The Isomap plot results: (a) using only weekdays' data for training, (b) using weekdays' and Sunday's data for training, and (c) using weekdays' and the whole weekend's data for training.

4.2.3 Computation Time

All experiments were performed on regular Pentium-5 machine. All computation regarding to one view was done less than five seconds on average.

4.2.4 Discussion

We would like to discuss several issues related to the proposed method and the focused problem. Some possible future extensions to this work will also be mentioned.

- In this work, to obtain the final anomaly detection result we simply intersect two or more views for view association. We can however to have a more robust view association procedure. First, we need to have criteria to judge which view is trustworthy, such as in this case, we can assume the *flow* and the *speed* views more trustworthy than the *duration* view because the *flow* and the *speed* views are provided by the sensors directly and the *duration* view is a derived feature based on several sensor readings. Second, we can consider different types of view association. For instance, we can consider adjust the portion of anomalies in each view so that the intersection is maximized. It is one of our future research topics.
- We studied both of the *duration* view and the *speed* view. It seems that they may share some correlation between each other, hence not appropriate to be used simultaneously to detect anomalies. The *duration* feature is defined as dividing the distance of consecutive stations by the *speed*. In this case study, the distance of consecutive stations may not remain constant (ranged from a half to one mile); therefore, the *duration* and *speed* may not refer to the same concept.
- We use Isomap mainly for data representation and visualization. To perform data clustering, we can either use hierarchical clustering method or perform regular data clustering such as K-means in the dimensionality-reduced (intrinsic) space. The difference between them is that the hierarchical clustering gives relatively stable result, compared to, e.g., the one from K-means if applied to the intrinsic space.
- In our study, we apply the 90-10 principle to confirm whether or not a group of data are considered as anomalies. In this work, all the results satisfy this principle. However, we should consider some adjustment once the principle is not fitted to the clustering result. We can try the following options:
  - Still stick with the 90-10 principle, but we move the points from the major group or the minor group or vice versa for the points closest to the group boundaries until 90-10 rule is satisfied.
  - Let the hierarchical clustering result or the clustering result in the intrinsic space (based on Isomap) decides the result itself.
  - Apply another flexible principle such as cutting data into two portions p and $1-p$ where p is between 0 and 1.

5. CONCLUSION

We proposed a method to detect anomalies in ITS data for traffic analysis. Different from previous anomaly detection approaches, we mainly focused on automatically generating an anomaly report, in this case, the incident report that shall

be helpful for drivers and police to act accordingly for the incident. In this report, the incident location, the moment of incident, and more importantly how the incident affects the traffic, for how long are all included based on our detection result. Therefore the police patrol can decide the significance of the incident and make appropriate judgment about whether or not they should go to the incident location for investigation and recovery and which incident they should take care first if more than one incident occurs at similar moments. Regarding to the methodology, we detect anomalies based on a view association approach given the multi-view information where each single view is used to detect anomalies, and the results from different views are combined for the final detection result. The method has many benefits if compared to previous anomaly detectors: 1) it needs little parameter tuning; 2) it needs no clean data training as the initial step; 3) it can work efficiently such as the algorithm can easily implemented in a parallel fashion. We evaluated the proposed method on *Mobile Century* and *PeMS* datasets. The evaluation shows the proposed method is effective at detecting incidents from the data. Even though we focused only on anomaly detection in ITS data in this work, it would not be surprising if the proposed method is easily generalized to other types of applications where anomaly detection is necessary. Investigating the above possibility is one of our future research plans.

6. REFERENCES

[1] A. Agovic, A. Banerjee, A. Ganguly, and V. Protopopescu. Anomaly detection using manifold embedding and its applications in transportation corridors. *Intell. Data Anal.*, 13(3):435–455, Aug. 2009.

[2] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *COLT: Proceedings of the Workshop on Computational Learning Theory, Morgan Kaufmann Publishers*, pages 92–100, 1998.

[3] S.-L. Chang, L.-S. Chen, Y.-C. Chung, and S.-W. Chen. Automatic license plate recognition. *Intelligent Transportation Systems, IEEE Transactions on*, 5(1):42–53, 2004.

[4] S. Chawla, Y. Zheng, and J. Hu. Inferring the root cause in road traffic anomalies. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, pages 141–150. IEEE, 2012.

[5] C. Chen, D. Zhang, P. S. Castro, N. Li, L. Sun, and S. Li. Real-time detection of anomalous taxi trajectories from GPS traces. In *Mobile and Ubiquitous Systems: Computing, Networking, and Services*, pages 63–74. Springer, 2012.

[6] T. F. Cox and M. A. A. Cox. *Multidimensional Scalling, Second Edition*. Chapman & Hall/CRC, 2000.

[7] S. Forrest, S. A. Hofmeyr, A. Somayaji, and T. A. Longstaff. A sense of self for unix processes. In *Security and Privacy, 1996. Proceedings., 1996 IEEE Symposium on*, pages 120–128. IEEE, 1996.

[8] Z. Fu, W. Hu, and T. Tan. Similarity based vehicle trajectory clustering and anomaly detection. In *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, volume 2, pages II–602. IEEE, 2005.

[9] J. C. Herrera, D. B. Work, R. Herring, X. J. Ban, Q. Jacobson, and A. M. Bayen. Evaluation of traffic data obtained via GPS-enabled mobile phones: The mobile century field experiment. *Transportation Research Part C: Emerging Technologies*, 18(4):568–583, 2010.

[10] F. Jiang, Y. Wu, and A. K. Katsaggelos. A dynamic hierarchical clustering method for trajectory-based unusual video event detection. *Image Processing, IEEE Transactions on*, 18(4):907–913, 2009.

[11] I. T. Jolliffe. *Principal component analysis*, volume 487. Springer-Verlag New York, 1986.

[12] A. Kind, M. P. Stoecklin, and X. Dimitropoulos. Histogram-based traffic anomaly detection. *Network and Service Management, IEEE Transactions on*, 6(2):110–121, 2009.

[13] P. G. Michalopoulos, R. D. Jacobson, C. A. Anderson, and T. B. DeBruycker. Automatic incident detection through video image processing. *Traffic engineering and control*, 34(2), 1993.

[14] P. Mirchandani and L. Head. A real-time traffic signal control system: architecture, algorithms, and analysis. *Transportation Research Part C: Emerging Technologies*, 9(6):415–432, 2001.

[15] C. Piciarelli and G. Foresti. On-line trajectory clustering for anomalous events detection. *Pattern Recognition Letters*, 27(15):1835–1842, 2006.

[16] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[17] H. Ringberg, A. Soule, J. Rexford, and C. Diot. Sensitivity of pca for traffic anomaly detection. *ACM SIGMETRICS Performance Evaluation Review*, 35(1):109–120, 2007.

[18] P. H. A. Sneath. *Numerical taxonomy : the principles and practice of numerical classification*. W.H. Freeman, San Francisco, 1973.

[19] J. B. Tenenbaum, V. De Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[20] S. Thajchayapong and J. A. Barria. Anomaly detection using microscopic traffic variables on freeway segments. *CD-ROM, Transportation Research Board of the National Academies*, 2010.

[21] D. Valerio, T. Witek, F. Ricciato, R. Pilz, and W. Wiedermann. Road traffic estimation from cellular network monitoring: a hands-on investigation. In *Personal, Indoor and Mobile Radio Communications, 2009 IEEE 20th International Symposium on*, pages 3035–3039. IEEE, 2009.

[22] J. Zhang, F.-Y. Wang, K. Wang, W.-H. Lin, X. Xu, and C. Chen. Data-driven intelligent transportation systems: A survey. *Intelligent Transportation Systems, IEEE Transactions on*, 12(4):1624–1639, 2011.

[23] Z. Zhang, K. Huang, and T. Tan. Comparison of similarity measures for trajectory clustering in outdoor surveillance scenes. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 3, pages 1135–1138. IEEE, 2006.

On-line Relevant Anomaly Detection in the Twitter Stream: An Efficient Bursty Keyword Detection Model

Jheser Guzman
PRISMA Research Group
Department of Computer Science
University of Chile
Santiago, Chile
jguzman@dcc.uchile.cl

Barbara Poblete
PRISMA Research Group
Department of Computer Science
University of Chile
Yahoo! Labs Santiago
Santiago, Chile
bpoblete@dcc.uchile.cl

ABSTRACT

On-line social networks have become a massive communication and information channel for users world-wide. In particular, the microblogging platform Twitter, is characterized by short-text message exchanges at extremely high rates. In this type of scenario, the detection of emerging topics in text streams becomes an important research area, essential for identifying relevant new conversation topics, such as breaking news and trends. Although emerging topic detection in text is a well established research area, its application to large volumes of streaming text data is quite novel. Making *scalability*, *efficiency* and *rapidness*, the key aspects for any emerging topic detection algorithm in this type of environment.

Our research addresses the aforementioned problem by focusing on detecting significant and unusual bursts in keyword arrival rates or *bursty keywords*. We propose a scalable and fast on-line method that uses normalized individual frequency signals per term and a windowing variation technique. This method reports keyword bursts which can be composed of single or multiple terms, ranked according to their importance. The average complexity of our method is $O(n \log n)$, where n is the number of messages in the time window. This complexity allows our approach to be scalable for large streaming datasets. If bursts are only detected and not ranked, the algorithm remains with lineal complexity $O(n)$, making it the fastest in comparison to the current state-of-the-art. We validate our approach by comparing our performance to similar systems using the TREC Tweet 2011 Challenge tweets, obtaining 91% of matches with LDA, an off-line gold standard used in similar evaluations. In addition, we study Twitter messages related to the SuperBowl football events in 2011 and 2013.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Retrieval models, Information filtering*

Keywords

Twitter, Text Mining, Bursty Keyword Detection

ODD'13, August 11th, 2013, Chicago, IL, USA.

1. INTRODUCTION

Social media microblogging platforms, such as Twitter[1], are characterized by extremely high exchange rates. This quality, as well as its network structure, makes Twitter ideal for fast dissemination of information, such as breaking news. Furthermore, Twitter is identified as the first media source in which important news is posted [10] and national disasters (e.g. earthquakes, diseases outbreaks) are reported [6].

Each message that is posted on Twitter is called a *tweet*, and messages are at most 140-characters long. In addition, Twitter users connect to each other using a follower/followee directed graph structure. Users can support and propagate messages by using a *re-tweet* feature which boosts the original message by reposting it to the user's followers.

Given that Twitter has been adopted as a preferred source for instant news, *real-time detection* of emerging events and topics has become one of the priorities in on-line social network analysis. Research on emerging topic detection in text is now focused on the analysis of *streaming data* and *on-line identification* [1]. In particular, the analysis of microblog text is quite challenging, mostly because of the large volume and high arrival rate of new data. In this scenario an important part of the analysis must be performed in real-time (or close to real-time), requiring *efficient* and *scalable* algorithms.

This problem has generated increasing interest from the private and academic communities. New models are constantly being developed to better understand human behavior based on social media data in interdisciplinary work fields such as sociology, political science, economy and business markets [3, 26].

We target the problem of emerging topic detection in short-text streams by proposing an algorithm for *on-line Bursty Keyword Detection (BD)* (definitions in Figure 1). This in general is considered to be a first important step in the identification of emerging topics in this context. Our approach uses windows slicing and window relevance variation rate analysis on keywords.

We validate our methodology on the TREC Tweet 2011 dataset, using it to simulate streaming data. Our experiments indicate that this concept works better at detecting keyword bursts in text streams than other more complex state-of-the-art solutions based on queue theory [8, 14, 24]. Also we analyze tweets from the SuperBowl football events in 2011 and 2013 as case studies. We perform a detailed discussion on the behavior of noise in bursty keyword signals which is constituted mostly by stopwords. We compare our solution to LDA [2] as a ground truth, similarly to prior work [24]. Using LDA in a one-keyword per topic mode, we achieved more

[1] http://www.twitter.com

than 91% topic matches. This is a great improvement over similar approaches. Moreover, our implementation is efficient, achieving a complexity of $O(n \log n)$ in the average case, where n is the number of messages in the current window.

In detail, the contributions of our work are three-fold:

1. We introduce a scalable and efficient keyword burst detection algorithm for microblog text streams, based on window slicing and relevance window variations.
2. We present a technique for eliminating non-informative and irrelevant words from microblog text streams.
3. We present a detailed proof-of-concept system and validate it on a public dataset.

Definition 1: A *keyword* is an informative word used in an information retrieval system to indicate the content of a document.
Definition 2: A *bursty keyword* is defined as a word or set of words that suddenly appear in a text stream at an unusually high rate [14].
Definition 3: A *topic* [*noun*] is a discussion or conversation subject. In other words, it is a set of keywords with semantic association.
Definition 4: An *event* is a topic which in general, corresponds to a real-world occurrence. It is usually associated with a geographic location and a time.
Definition 5: The concept of *relevance* used in this work means: the representation of a term in a window. In other words "the probability of occurrence of a term in a Window".

Figure 1: Definitions used in the paper (Oxford Dictionary)

This paper is organized as follows. Section 2 presents an overview of relevant literature for this work. Section 3 presents a complete description of our proposal, divided in two parts: the burst detection algorithm and our proof-of-concept system. Section 4 summarizes our experimental validation and results on the SuperBowl 2011 and SuperBowl 2013 events. Section 5 delivers our conclusions and discuss future work.

2. RELATED WORK

Our work involves research in the areas of event detection and trend analysis for microblog text streams. In particular our goal is to *identify current popular candidate events listed by most popular bursty terms in the data stream*. In relation to this topic, several applications exist for detecting events like natural disasters and health alerts. For example, epidemics [11], wildfires [23], hurricanes and floods [20], earthquakes and tornados [7, 17].

Events have been modeled and analyzed over time using keyword graphs [18], link-based topic models [13], and infinite state automatons [8]. Leskovec *et al.* [12] perform analyses of memes for news stories over blogs and news data, and on Twitter data in [25]. Swan *et al.* [21, 22] deal with constructing overview timelines of a set of news stories.

On-line bursty keyword detection is considered as the basis for on-line emerging topic detection [14]. For topic detection in an off-line fashion, algorithms such as Latent Dirichlet Allocation (LDA) [2] or Phrase Graph Generation Method [19] can be used. Statistical methods and data distribution tests can also be used to detect bursty keywords [8].

Mathioudakis and Koudas introduce Twitter Monitor [14], a system that performs detection of topic trends (emerging topics) in the Twitter stream. Trends are identified based on *individual keyword* bursts and are detected in two steps. This system identifies bursts of keywords by computing the occurrence of individual keywords in tweets. The system groups keyword trends based on their co-occurrences. To detect bursts of keywords, they introduce the algorithm QueueBurst with the following characteristics: (1) *one-step analysis per keyword*, (2) *real-time analysis (based on the tweet stream)*, (3) *adjustable against "false" explosions*, and (4) *adjustable against spam*. In order to group sets of related bursty keywords, the authors introduce an algorithm named GroupBurst, which evaluates co-occurrences in recent tweets. Our current work focuses on keyword detection up to three term keywords, for which we compare to QueueBurst in Section 4. Twitter Monitor requires an intensive pre-processing step for determining its optimal parameter settings for each keyword and also for global variables. These parameter settings must be computed with a historical dataset.

A different approach is presented by Weng *et al.* [24], with their system EDCoW (Event Detection with Clustering of Wavelet-based Signals). EDCoW builds individual word signals by applying wavelet analysis to word frequencies. It filters away trivial words by looking at their corresponding signal auto-correlations. The remaining words are clustered to form events with a modularity-based graph partitioning technique. The wavelet transform is applied to a signal (time-series) created using the TF-IDF index [5]. Their approach was implemented in a proof-of-concept system, which they used to analyze online discussions about the Singapore General Election of 2011.

Another relevant study is that of Naaman *et al.* [15]. In this work the authors make two contributions for interpreting emerging temporal trends. First, they develop a taxonomy of trends found in data, based on a large Twitter message dataset. Secondly, they identify important features by which trends can be categorized, as well as the key features for each category. The dataset used by Naaman *et al.* consists of over 48 million messages posted on Twitter between September 2009 and March 2010 by 855,000 unique New York users. For each tweet in this dataset, they recorded its textual content, the associated timestamp and the user ID.

3. BURST DETECTION MODEL

We propose a methodology based on time-window analysis. We compute keyword frequencies, normalize by relevance and compare them in adjacent time windows. This comparison consists of analyzing variations in *term arrival rates* and their respective *variation percentages* per window. A similar notion (Discrete Second Derivative) has been used in the context of detection of bursts in academic citations [4]. We define *Relevance Rates (RR)* as the probability of occurrence of a non-stopword term in a window. We use RR to generalize burst detection making them independent of the arrival rate. Even though the public Twitter API only provides a stream which is said to be less than 10% of the actual tweets posted on Twitter, we believe our method can be easily adapted for the complete data stream using a MapReduce schema [see Section 4.2.3]. Arrival rates vary periodically (in a non-bursty way) during the day; depending on the hour, time zone, user region, global events and language (shown in Figure 4).

Bursty keywords are ranked according to their *relevance variation rate*. Our method avoids the use of statistical distribution analysis methods for keyword frequencies; The main reason is that this approach, commonly used in state-of-the-art approaches, increases the complexity of the process. We show that a simple relevance variation concept is sufficient for our purposes if we use good stopword filter and noise minimization analysis [see section 4.1.1].

To study the efficiency of our algorithm, which we name *Window Variation Keyword Burst Detection*, we implement a proof-of-concept system. The processes involved in this system are five. These modules are independent of each other and they have been structured for processing in threads. These modules are: Stream

Listener, Tweets Filterer, Tweets Packing, Window Processing and Keyword Ranker. This architecture allows us to process all the input data with linear complexity, making it scalable for on-line processing.

3.1 Data Pre-Processing

In this stage, data is pre-processed extracting keywords from each message, so that later on, our burst detection algorithm analyzes them (see Figure 2 and 3). This stage is composed of the following three modules: *(1) Stream Listener*, *(2) Tweet Filter* and *(3) Tweet Packing*.

1) Stream Listener Module: This module receives streaming data in the form of Twitter messages, which can come directly form the Twitter API or some other source[2] . Messages are received in JSON format[3]. This data is parsed and encapsulated. After the encapsulation of each message it is en-queued in memory for the next module in the pipeline. It should be noted that message encapsulation is prone to delays caused by the Internet bandwidth connection and Twitter's information delivery rate, which can cause data loss.

2) Tweet Filter Module: This module discards messages which are not written in languages accepted by our system. We perform language classification using a Naive Bayes classifier. This module also standardizes tweets according to the following rules:

- Treatment of special characters and separation marks: Replacing special characters and removal of accents, apostrophes, etc.
- Standardization of data: Upper and lower case conversion and replacement of special characters.

After normalization and language detection, the tweet is enqueued into queue Q_1 for posterior analysis.

3) Tweet Packing Module: Filtered and standardized tweets in queue Q_1, are grouped into a common set determined by creation timestamp, shown in Figure 2. This set of tweets, which we refer to as *Bag of Tweets*, represent an individual time-window. It is important to note that the arrival of tweets maintains a chronological order. In the case that an old or delayed tweet appears, it is included in the current window. Each of these windows is sent to the following stage for processing.

3.2 Bursty Keyword Detection

This process involves two modules. The second module must wait for the first module to finish processing a window in order to process an entire window at a time (serial mode). The algorithm shown in Figure 3 describes this process, where the second module starts in line 24.

1) Window Processing Module: Each keyword, composed of a single or adjacent word n-grams, is mapped into a hash table data structure. This structure manages keywords in addition to the information of its two adjacent windows and their rates. We consider as *n-grams* the n ordered correlative words.

The hash table allows access to keyword information in constant time for most cases *O(1)*, and in the worst case with complexity of *O(n)* when collisions occur. This process is detailed in the algorithm described in Figure 3. This data structure controls the complexity of the algorithm with optimal insertions and search $O(1)$.

2) Keyword Ranker Module: Bursty keywords are included implicitly into the hash table. Therefore, we extract bursty keywords by discarding those that do not classify as having a positive relevance variation. We discard non-bursty keywords using the criteria

[2] `http://twitter4j.org/`

[3] `https://dev.twitter.com/docs/tweet-entities`

Require: Global Queue Q_1 of previously filtered tweets, and Global Queue Q_2 of keyword bags. $window\_time$ is the earliest timestamp of the tweets in the same Bag.
Consider $window\_size$ as the time length of the window.

1: $initTime \Leftarrow null$
2: **while** $t' \Leftarrow get\_tweet\_from\_queue(Q_1)$ **do**
3: **if** $initTime = null$ **then**
4: $initTime \Leftarrow timestamp(t')$
5: $endTime \Leftarrow initTime + window\_size$
6: $TBag \Leftarrow \phi$
7: Set $initTime$ to $window\_time$ in $TBag$
8: **end if**
9: **if** $timestamp(t') < endTime$ **then**
10: $t'' \Leftarrow filter\_stopwords(t')$
11: $TBag \Leftarrow TBag \bigcup keywords(t'') \bigcup word\_nGrams(t'')$
12: **else**
13: $put(TBag)$ in Q_2
14: Create new $TBag$
15: $TBag \Leftarrow \phi$
16: Set $endTime$ to $window\_time$ in $TBag$
17: $initTime \Leftarrow endTime$
18: $endTime \Leftarrow initTime + window\_size$
19: **end if**
20: **end while**

Figure 2: Packer Thread

Require: Global Queue Q_2 of keyword bags, and a Global hash table HT mapping $< Keyword, < window_1, window_2, rates >>$

1: **while** $TBag \Leftarrow get\_bag\_from\_queue(Q_2)$ **do**
2: $TotalWords \Leftarrow size(TBag)$
3: **for all** $word \in TBag$ **do**
4: **if** $word \notin mapped\_keywords(HT)$ **then**
5: $window\_time \Leftarrow window\_time(TBag)$
6: $freq \Leftarrow 0$
7: $window \Leftarrow < window\_time, freq >$ {create a new window}
8: $put\_in\_hashtable\_as\_window1(word, window)$
9: **else**
10: $window \Leftarrow get\_from\_hashtable(word, HT)$
11: $w\_time \Leftarrow window\_time(window)$
12: $b\_time \Leftarrow window\_time(TBag)$
13: **if** $w\_time \neq b\_time$ **then**
14: $window_2 \Leftarrow window_1$ for $word$ in HT
15: $window\_time \Leftarrow window\_time(TBag)$
16: $freq \Leftarrow 0$
17: $window \Leftarrow < window\_time, freq >$ {create a new window}
18: $put\_in\_hashtable\_as\_window1(word, window)$
19: **end if**
20: Add 1 to $freq$ in $window_1$ for $word$ mapped in HT
21: **end if**
22: **end for**
23: $set\_relevance\_for\_each\_keyword\_with(TotalWords)$
24: $WordRank \Leftarrow \phi$
25: **for all** $mword \in mapped\_keywords(HT)$ **do**
26: $window_1 \Leftarrow get\_window_1(mword, HT)$
27: $window_2 \Leftarrow get\_window_2(mword, HT)$
28: $relevance_1 \Leftarrow get\_relevance(window_1)$
29: $relevance_2 \Leftarrow get\_relevance(window_2)$
30: **if** $(relevance_1 - relenvance_2) > 0$ **then**
31: $WordRank \Leftarrow WordRank \bigcup mword$
32: **end if**
33: **end for**
34: $WordRank' \Leftarrow quickSort(WordRank)$
35: display $WordRank'$
36: **end while**

Figure 3: Processor Thread

Table 1: Tweet language classification results

Language	%	Freq.
English	34.4	4,287,605
Romanic (Neo-Lat)	20.9	2,599,849
German	2.5	305,228
Others	42.2	5,253,562
TOTAL	100.0	12,446,244

Table 2: Average word count per message

Language	Words per Msg
English	7
Romanic (Neo-Lat)	12
German	10

described below, this prevents the size of the hash table from growing out of control:

1. It is the first occurrence of the keyword. We must wait until the next window to check it again.
2. We observe a negative variation in frequency rates between adjacent windows.
3. Low arrival rate: Many words do not appear frequently. We discard these words if the average arrival rate is lower than 1.0 (keywords per time-window).

The remaining keywords are sorted in descending order according to their *Relevance Variation Rate*. Keywords with the highest variation rate or *burstiness* are ranked in the top positions.

4. EVALUATION

In this section we explain empirical parameter settings for our system and perform a validation by comparing it against state-of-the-art methods. For evaluation and comparison purposes we adapt the evaluation used in [24] with LDA as a gold standard. LDA is used to process data off-line and generate topics listed by its most likely keyword.

4.1 Dataset Description

The datasets used in this experiment are the TREC Tweet 2011 Challenge dataset from the National Institute of Standards and Technology (NIST[4]): In addition, we use tweets related to the Super-Bowl 2013 football event obtained using Twitter API on February 3rd. As part of the TREC 2011 microblog track, Twitter provided identifiers for approximately 16 million tweets, sampled from its full data-stream between January 23rd and February 8th, 2011. The corpus is designed to be a reusable and representative sample of the Twitter stream. The TREC Tweet 2011 messages are obtained using a crawler, fed with tweet IDs provided by NIST for downloading messages directly from Twitter. As expected, not all messages were obtained in the downloading process, mostly because of: the tweet having been removed, transmission or server errors, or changes made in the access permissions by the owner of the message. Therefore, we only obtained $12,446,244$ tweets.

The origin of these messages is random regarding their geographic location, and their languages. To identify the language of each message, we use the java library *LangDetect*[5] which claims 93% accuracy in the detection of 59 languages in short messages[6]. Using this classifier we obtain the classification shown in Table 1.

In this work we only use English, Neo-Lat[7] and German languages. These correspond to 57.8% of the dataset with a total of $7,192,682$ messages. The main reasons for selecting these languages are:

- The use of the *space character* as a word delimiter.
- Same writing structure order (left to right and top to bottom).
- Same set of character symbols for writing.

In table 2 we show the average number of words per message in each language. Twitter's 140-character limit for messages is an advantage for the filtering and tokenizing steps which take place in lineal complexity $O(wn)$ using single words, and $O(w^2n)$ adding adjacent word n-grams. The variable n corresponds to the number of tweets to be processed and w the average number of words per message. Since w is a constant upper-bound, we conclude that the complexity of the algorithm described in Figure 2 remains lineal $O(n)$.

4.1.1 Stopword analysis

We consider *stopwords* as words used to connect objects, actions and subjects. Stopwords are omitted in our experiments (see Figure 2, line 10) and discarded since they do not add semantic value or represent events.

The *Window Variation Keyword Burst Detection* (BD) technique computes keyword arrival frequency in time-windows using a predefined window length that remains constant during the entire process. The use of the absolute frequency value is the most intuitive approach, but it does not work correctly because the arrival rate is not constant during the day. This is explained in correlation with sleep hours, work hours, days of the week or season of the year in which the tweet is posted (many of these changes can show periodicity in time). Other factors can affect tweet arrival rate behavior, with an important one being language and geographical location of the user at that moment. As shown in Figure 4, frequency signals decrease between 2:00 and 9:00 GMT hours, especially during sleeping hours. The signal slowly increases between 9:00 and 17:00 GMT hours, when people are at work. After that, the signal remains stable until 2:00 GMT the next day. Therefore, we opt for using *Relevance Variation rates* for window comparison, which creates independence between values and the hour of the day.

In Figure 4 we show frequency signals according to the arrival rate of each language. We observe that country time zone can affect the behavior of certain keywords. Portuguese tweets are clearly shifted some hours in comparison to other languages. We believe that this happens because a large number of tweets in this language originate in Brazil. Again, this effect is mitigated by the use of relative rates (or percentages) instead of absolute values.

A common technique to filter or discard stopwords is the use of predetermined list of stopwords per languages. In social media, especially in Twitter, the behavior of stopwords is dynamic. This occurs because the message limit of 140-characters forces the user to reduce or transform their message to utilize less space. Therefore, stopwords in Twitter messages vary from those of standard document lists. To identify stopwords for our experiment in this type of environment (microblogging), we download standard stopword lists[8] and map them onto a frequency histogram of our keyword

[4] `http://trec.nist.gov/data/tweets/`

[5] `http://code.google.com/p/language-detection`

[6] http://shuyo.wordpress.com/2011/11/28/language-detection-supported-17-language-profiles-for-short-messages/

[7] Romanic and Romance Languages: Spanish, French, Italian, Portuguese.

[8] Stopwords List: `http://snowball.tartarus.org/algorithms/`

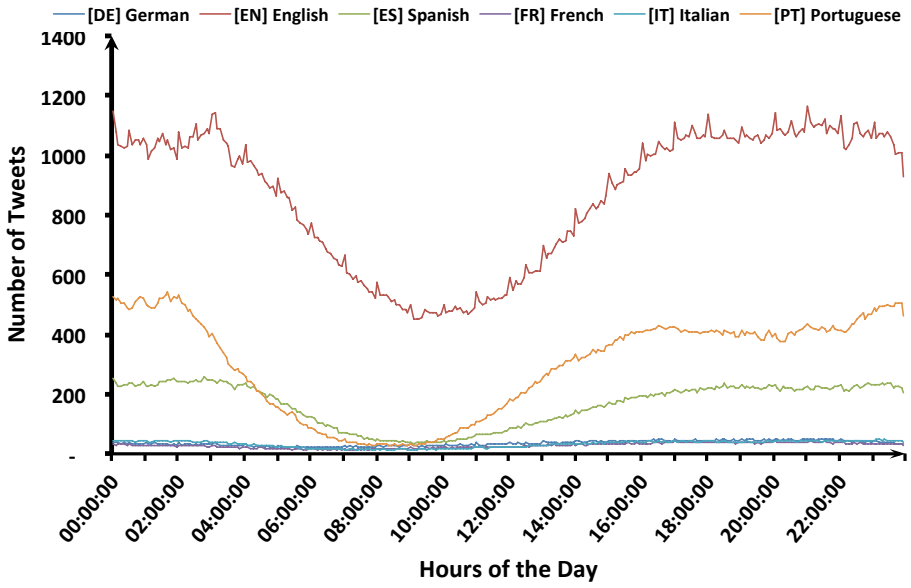


Figure 4: Tweet arrival rate per language (GMT timezone): Shows differences between language volumes and the shift in the signals due to different timezones.

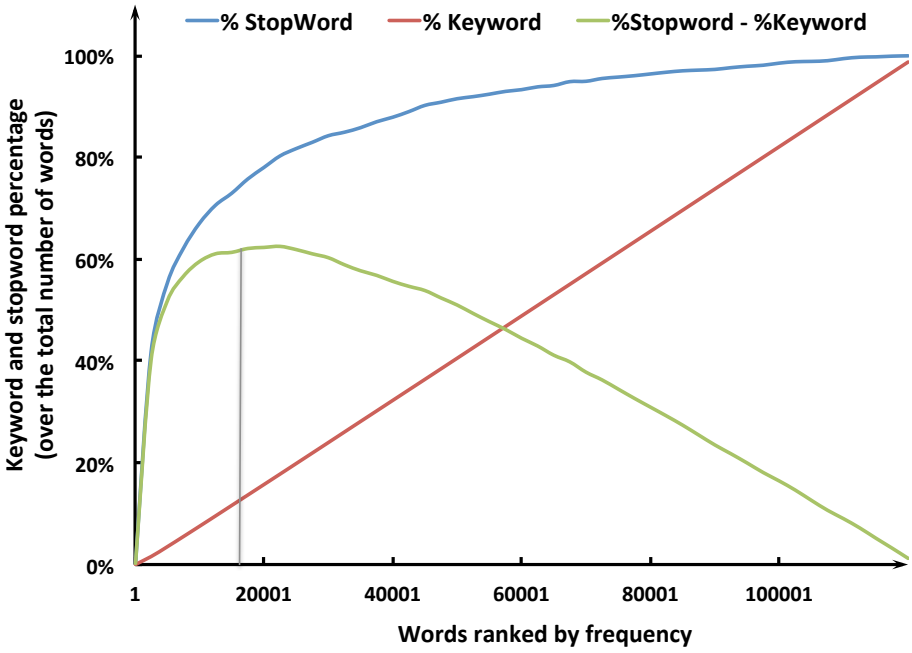


Figure 5: Keywords and stopwords ranked by frequency histogram (accumulative): The exact point where stopwords start decreasing their volume.

dataset (made up of the complete vocabulary found in our collection of tweets). We observe that most stopwords are concentrated in the area of the histogram which contains words with a high arrival rate. Therefore, we mark as candidate stopwords all keywords with more than $18,000$ frequency occurrence in our keyword data set. Nevertheless, by inspecting our stopword candidate list we see that it still contains some relevant keywords which we do not wish to mark as stopwords (false negatives). These cases occur because some relevant keywords have high occurrence rates during certain time periods. In order to avoid these cases, we compute the *Relative Standard Deviation* (RSD) rate[9] and use it to describe frequency signal stability of keywords. Intuitively, stopword signals should be noisier (i.e., have a higher $standard\_deviation$) than the signal of regular keywords because they do not represent an organized phenomena. Figure 6 shows that stopwords are **twice as noisy as regular words**, therefore we consider this as a parameter for stopword identification. Also, we measure RSD using different window sizes (shown in Figure 6) in order to identify the optimal *window size* for which stopwords and keywords are less affected by noise.

Using the previously described methodology, we obtain a final list of stopwords that contains $18,631$ words. With this, we remove 90% of total word arrivals, making the process **significantly faster**. It also **reduces the volume of words** for the final hash table in algorithm in Figure 3, therefore requiring less memory. Stopwords can be updated by performing off-line batch analysis which does not affect on-line performance.

4.1.2 Determining optimal window size

The stability of the signal (minimization of RSD) helped us determine the optimal window size for our algorithm. This is a crucial parameter that determines the performance of our solution.

- If the window size is too small, the occurrence of empty windows for a term increases (frequency equal to 0), making the noise rate increase and frequency rate tend to zero.
- On the other hand, if the window size is too large, the stability of the signal becomes constant and bursty keyword detection is delayed.

Therefore, it seems reasonable to place optimal window size somewhere between 17 minutes and 2 hours. In practice, we select windows of 20 minutes for actual tweet arrival rate for *fast detection* of bursts. This choice is practical because it divides a 24-hour day exactly, making the analysis easier to understand and to compare windows from different days. It is important to remark that this decision of 20 minutes is for the actual Twitter API arrival rate of tweets that represent less than 10% of the total system. We expect to reduce the time of the window size if the arrival rate is higher. The effect of using higher size on windows would not detect bursts because the signal will tend to the average value. Using 20 minutes as window size, we minimize the time of bursts detection keeping the noise level in the signal to a minimum.

4.2 Comparison to other methods

TwitterMonitor (TM) [14] is one of the earliest works in the field of detecting emerging topics on Twitter. Its core algorithm named QueueBurst uses five parameters and concepts of Queue Theory M/M/1. We used the recommended parameter settings from the technical paper provided directly to us by the authors, setting the tolerances ϵ_0 and ϵ_1 to 10^{-2}, and the ratio r to 2. The arrival rates (λ_0) per each keyword were calculated using the first week of tweets of the TREC Tweet 2011 dataset to set each keyword.

[9]Relative Standard Deviation (RSD): $Standard\_Deviation/Average$

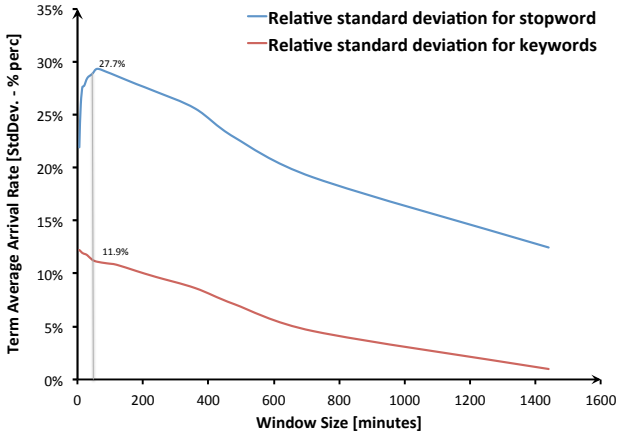


Figure 6: Relative standard deviation of keywords and stopwords: Noise in stopwords is higher than in keywords (non-stopwords). The 20-minute windows size stabilizes the noise in both signals keeping the window size as short as possible.

Weng *et al.* [24] developed EDCoW (Event Detection with Clustering of Wavelet-based Signals), a system that uses queueing technique for bursty keyword detection [8] and wavelet techniques to detect trends in Twitter.

As mentioned earlier, LDA (with the Gibbs Sampling technique for parameter estimation [16]) is a reasonable gold standard for our evaluation. This follows the approach used in the EDCoW article [24] to compare TM and EDCoW with our proposal. Given that with EDCoW there are no details available for the implementation, we can only perform a similar experiment and compare results to TM and our method.

Next, we analyze the complexity of TM, LDA and BD. We cannot analyze the complexity of EDCoW. It should be noted that LDA is an off-line method, therefore it competes with an advantage in relation to on-line methods TM, BD (and EDCoW) given that it has a complete view of the information, as opposed to the limited data that on-line methods use.

- TM is $O(wni)$, where n is the number of tweets to be processed, w is the average number of words per message and i is the iterations for generating the exponential aleatory variables for the M/M/1 queue. The parameter i cannot be determined exactly because of randomness, leaving the complexity of the algorithm in $O(ni)$. This algorithm analyzes each tweet in one pass, but this property creates delays because it cannot be parallelized.
- LDA is a statistical method that determines k topics that represent a document, where k is the number of the top events in that document. Assume we have N documents and a vocabulary of size V. The complexity of mean-field variational inference for LDA is $O(NKV)$. In this scenario, we assume a document to be the concatenation of all of the tweets in a same time-window (as in [24]). It should be noted that LDA does not constitute a perfect gold standard for burst detection in streaming data, but constitutes an approximation. Some topics might not be bursty and some bursts do not correspond to topics.
- For our *Window Variation Keyword Burst Detection* (BD) a threshold T can be included in order to truncate the number of bursty keywords returned. This, and an optimal selection algorithm [9], reduce complexity of the ranking algorithm to

Table 3: Examples of bursty keywords detected in the TREC Tweet 2011 dataset for February 6th, 2011

Window Time	Top 10 Keywords	Tweet Examples
17:20 GMT	liverpool, torres, justinbieber, meireles, chelsea, super, greenandyellow, blackandyellow, commercials, bowl	• berapa" chelsea liverpool? • A little over 6 hours until Super Bowl starts! Do you have everything you need? • #BrandBowl - Mullen Leverages @Twitter, @Radian6 to Rank the Best Super Bowl Ads http://cot.ag/hkL6H2 #sb45 • if torres is worth £50m I must be worth about a tenner . Nice one @LeedsLadAdam
20:40 GMT	corinthians, palmeiras, pra, julio, agora, jogo, chupa, gol, cesar, vai	• @ferpetucco palmeiras perdeu pro Corinthians ? O_O • 2T 37m. GOOOOOOOOOOOOOOOOLLLLLLLLLLLLL E DO CORINTHIANS!!!! • GOOOOOOOOOOL DO CORINTHIANS! Aleshow faz o dele! Palmeiras 0 x 1 Corinthians. #vccomenta
23:20 GMT	christina, superbowl, anthem, national, aguilera, sing, super, bowl, lea, singing	• National anthem over-under is 1:54. I think Christina Aguilera s hitting the over. • The Superbowl seems to be 90% entertainment, and 10% of the actual game. #bbcsuperbowl • superbowl lea michele cantandoooo :) • Que elegante se ve Christina, yo pense que iba a salir en minifalda con tanga :P.....:O

$O(Tn)$. Because T is constant it remains lineal $O(n)$. This technique does not require parameters to be reset at run time. Otherwise, if we decide not to use a threshold, the complete ranking would take $O(n\log(n))$.

We compare our BD algorithm to the topics returned by LDA on a one-keyword-per-topic basis. We do this to determine the percentage of topic matches on the TREC Tweet 2011 dataset. We also compared Twitter Monitor with LDA using the same dataset and assumptions, and discuss results obtained similarly in EDCoW [24].

4.2.1 Results and discussion

The comparison of BD and TM against LDA is performed window by window. We compare the number of keywords that overlap between each time-window and their respective match percentages (see Figure 7). Our system BD displays 91% coincidences with LDA. Comparing TM with LDA, there is only a 3% keyword match. We also looked at the overlap between bursty keywords reported by BD and TM, obtaining only a 14% match.

We believe that the low percentage of the coincidences between LDA and TM corresponds to the sensitivity of TM's empirical parameter settings. Also, TM keeps reporting the same keywords in time because they have not yet been dropped in the following windows (while they do not satisfy the hypothesis H_0 of the TM algorithm).

In addition, the results reported by Weng *et al.* [24] for their system EDCoW are of a 22% match with LDA in the best case.

Therefore, in our experimental evaluation we observed that our algorithm (BD) outperformed TM and EDCoW with 91% coincidences with LDA versus 3% and 22% of the other methods respectively. Figure 7 shows the percentage of coincidences against LDA on the TREC Tweet 2011 Dataset. These percentages were estimated for each 20-minute time-window in the dataset. The horizontal-axis considers the beginning of the first window on January 23rd 00:00 hours and that of the last window on February 8th 23:40 hours.

It should be noted that we acknowledge that LDA is not a perfect gold standard for burstiness detection. Nevertheless, we believe that it constitutes a reasonable approximation. In particular, LDA detects topics and not all topics text are bursty. For example, if a topic is constant through the entire dataset, then it is not bursty (i.e. fan conversations of celebrities such as Justin Bieber). Also, not all bursts constitute topics, for example random hashtags which interconnect otherwise unrelated messages (i.e. the popular hashtag `#fail` is put at the end many messages which depict failed situations).

In addition, we study an interesting event on February 6th, 2011, due to the Super Bowl XLV. We observe that keywords related to this event rapidly reach the top positions in the *word rank* list (see Table 3). Interestingly, before the Super Bowl started, a soccer match between Corinthians and Palmeiras was played in Brazil, which is also shown in the top keywords. Also, keywords related to Chelsea and Liverpool soccer match displayed burstiness too. On February 3rd, 2013, the SuperBowl XLVII event occurs as well, and specific keywords related to the event appear highly ranked (Black-out, 49ers, Ravens, among others).

Table 3 and 4 show the top-10 keywords of a specific time-window, and a random sample of tweets from this time period containing some of these keywords. Note that the tweets do not include all of the keywords which are listed in the first column. In this example some tweets and keywords contain noise, but this is normal when important events occur (e.g massive sport matches, concerts, and other public events).We list keywords ranked according to highest burstiness.

4.2.2 Experiment Repeatability

We used a public dataset, openly available libraries and an easy to acquire computer to make the experiment repeatable. In this experiment we used a Home Personal Computer (PC) with Core2Quad Q6600 2.4Ghz Intel Processor with 4 cores, 8 GB in RAM using Linux Ubuntu x64 11.10 version as Operative System. The pro-

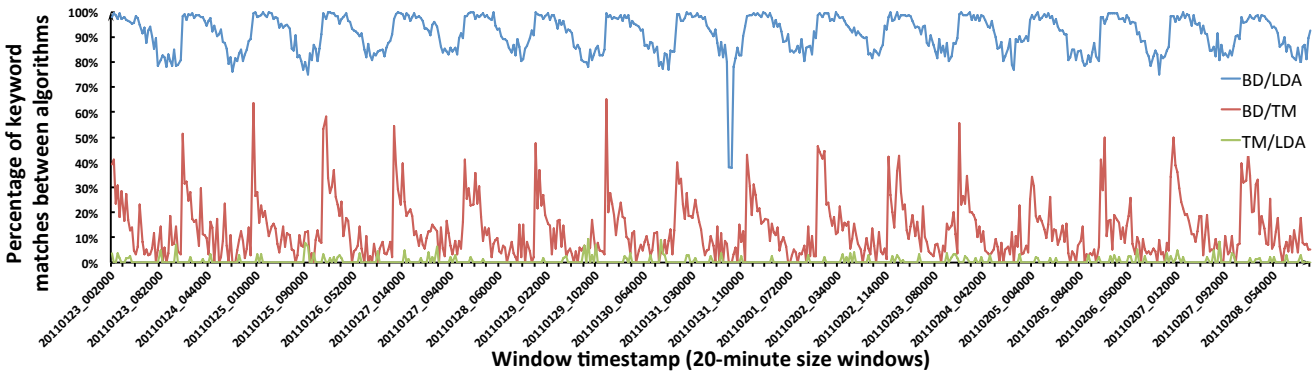


Figure 7: Comparison of the Window Variation Keyword Burst Detection algorithm with TwitterMonitor and LDA

Table 4: Examples of bursty keywords detected for the SuperBowl 2013 event, February 3rd and 4th, 2013

Window Time	Top 10 Keywords	Tweet Examples
Feb 3rd 21:20 GMT	ravens, beyonce, 49ers,#superbowlsunday, @mrbbutterfield, forward, hours, #superbowlxlvii, xlvii, halftime	• @ArianaGrande: @glitteryariana: @ArianaGrande 49ers or ravens? Beyonce lol made my day. • RT @NiallOfficial: Who do I follow in superbowl tonight american fans? 49ers or ravens? I don t know much about american football • 49ers: 27 Ravens: 24 Goes into quadruple OT. Monday declared national holiday.
Feb 3rd 23:40 GMT	ravens, touchdown, commercial, #thekiss, kaepernick, commercials, boldin, godaddy, tackles, @godaddy	• RT @lifestylist: #TheKiss? The Worst! @GoDaddy #NotBuyingIt. Wish they d spend that money on customer service instead of wasting it on a • RT @PiperJones: That GoDaddy commercial was just . . . ew.
Feb 4th 01:40 GMT	lights, power, outage, superdome, 49ers, ravens, turned off, stadium, luz, #lightsout	• RT @YourAnonNews: NFL can control everything except power in Superdome. What an embarrassment. 7 mins and counting. • RT @ShamelessProbs: The lights went out because we don t need football after Beyonce. • RT @billmarchi13: @TheShieldWWE turned off the lights and are making their way to take out #Flacco! @WWE #Superbowl

gramming language used to implement the algorithms was mainly Java. For implementing algorithm LDA we used the OpenSource GibbsLDA C++. The Dataset can be obtained upon request and must be acquired via NIST and downloaded using a crawler or an API on Twitter. We also will provide on request, code implementation of our algorithm for research purposes.

4.2.3 Scalability

After the process of a window by the Keyword Ranker Module, the entire system can be conceptually represented as a Node for a MapReduce Schema, using a specific term as a Key from merge data between other nodes in parallel processing. When we use various instances of the system as nodes, each one would have their own HashTable, and it is necessary to merge the sorted results of each one efficiently adding the frequencies of each repeated terms from all nodes, The computational cost of this procedure would take $O(n\log(n))$ using a variation of the merge step in MergeSort or much better a parallel variation of it. Once the hash tables data is merged, the method must recalculate the additional rates (arrival rate, relevance and variations). Thus, we will obtain a global result.

5. CONCLUSION

We have introduced a novel approach for on-line bursty keyword detection on the text streams. Our approach requires only the setting of the $window\_size$ parameter. It is efficient in the use of resources with a complexity of $O(n\log n)$ which makes the method *scalable*. This makes our approach easy to use and promising for on-line processing in comparison to other state-of-the-art methods. Experimental results indicate that our algorithm can scale to high tweet arrival rates while still producing high-quality results. Overall, our system produces an extraordinary keyword overlap against LDA, using very limited resources and memory.

In addition, we have presented an in-depth analysis of the behavior of *stopwords* on the Twitter stream and how to identify them. We also explain and justify the values for the parameters for our algorithm.

Future work is directed at identifying topics for fast and efficient detection of semantically coherent trends on Twitter.

Acknowledgements

The authors thank Michael Mathioudakis for his guidance in our implementation of the TwitterMonitor algorithm. We also thank Jorge Perez from the University of Chile for his valuable comments. Jheser Guzman was supported by CONICYT's Doctoral Program. Barbara Poblete was partially supported by FONDECYT grant 11121511 and Program U-INICIA VID 2012, grant U-INICIA 3/0612; University of Chile.

6. REFERENCES

[1] P. Adler and S. Kwon. Social capital: Prospects for a new concept. *Academy of management review*, pages 17–40, 2002.

[2] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet Allocation. *The Journal of Machine Learning Research*, 3:993–1022, 2003.

[3] J. Bollen, H. Mao, and X. Zeng. Twitter mood predicts the stock market. *Journal of Computational Science*, 2011.

[4] S. Gaito, M. Zignani, G. P. Rossi, A. Sala, X. Wang, H. Zheng, and B. Y. Zhao. On the Bursty Evolution of Online Social Networks. *arXiv preprint arXiv:1203.6744*, 2012.

[5] K. Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21, 1972.

[6] S. Kellert. *In the wake of chaos: Unpredictable order in dynamical systems*. University of Chicago press, 1993.

[7] K. Kireyev, L. Palen, and K. Anderson. Applications of topics models to analysis of disaster-related Twitter data. In *NIPS Workshop on Applications for Topic Models: Text and Beyond*, 2009.

[8] J. Kleinberg. Bursty and hierarchical structure in streams. *Data Mining and Knowledge Discovery*, 7(4):373–397, 2003.

[9] D. Knuth. The art ofcomputer programming, vol. 3: Sorting and Searching. *Reading, MA: Addison-Wesley*, 19:496–503, 1973.

[10] H. Kwak, C. Lee, H. Park, and S. Moon. What is Twitter? a social network or a news media? In *Proceedings of the 19th international conference on World Wide Web*, pages 591–600. ACM, 2010.

[11] V. Lampos, T. De Bie, and N. Cristianini. Flu detector-tracking epidemics on Twitter. *Machine Learning and Knowledge Discovery in Databases*, pages 599–602, 2010.

[12] J. Leskovec, L. Backstrom, and J. Kleinberg. Meme-tracking and the dynamics of the news cycle. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 497–506. ACM, 2009.

[13] C. Lin, B. Zhao, Q. Mei, and J. Han. Pet: a statistical model for popular events tracking in social communities. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 929–938. ACM, 2010.

[14] M. Mathioudakis and N. Koudas. Twittermonitor: Trend detection over the Twitter stream. In *Proceedings of the 2010 international conference on Management of data*, pages 1155–1158. ACM, 2010.

[15] M. Naaman, H. Becker, and L. Gravano. Hip and trendy: Characterizing emerging trends on Twitter. *Journal of the American Society for Information Science and Technology*, 2011.

[16] X. Phan and C. Nguyen. Gibbslda++, a C/C++ Implementation of Latent Dirichlet Allocation (LDA) using Gibbs Sampling for parameter estimation and inference. `http://gibbslda.sourceforge.net/`.

[17] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes Twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on World wide web*, pages 851–860. ACM, 2010.

[18] H. Sayyadi, M. Hurst, and A. Maykov. Event detection and tracking in social streams. In *Proceedings of International Conference on Weblogs and Social Media (ICWSM)*, 2009.

[19] B. Sharifi, M. Hutton, and J. Kalita. Summarizing microblogs automatically. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 685–688. Association for Computational Linguistics, 2010.

[20] K. Starbird, L. Palen, A. Hughes, and S. Vieweg. Chatter on the red: what hazards threat reveals about the social life of microblogged information. In *Proceedings of the 2010 ACM conference on Computer supported cooperative work*, pages 241–250. ACM, 2010.

[21] R. Swan and J. Allan. Extracting significant time varying features from text. In *Proceedings of the eighth international conference on Information and knowledge management*, pages 38–45. ACM, 1999.

[22] R. Swan and J. Allan. Automatic generation of overview timelines. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 49–56. ACM, 2000.

[23] S. Vieweg, A. Hughes, K. Starbird, and L. Palen. Microblogging during two natural hazards events: What Twitter may contribute to situational awareness. In *Proceedings of the 28th international conference on Human factors in computing systems*, pages 1079–1088. ACM, 2010.

[24] J. Weng, Y. Yao, E. Leonardi, and F. Lee. Event detection in Twitter. In *Fifth International AAAI Conference on Weblogs and Social Media*, 2011.

[25] J. Yang and J. Leskovec. Patterns of temporal variation in online media. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 177–186. ACM, 2011.

[26] X. Zhang, H. Fuehres, and P. Gloor. Predicting stock market indicators through Twitter, I hope it is not as bad as I fear. *Procedia-Social and Behavioral Sciences*, 26:55–62, 2011.

Distinguishing the Unexplainable from the Merely Unusual: Adding Explanations to Outliers to Discover and Detect Significant Complex Rare Events

Ted E. Senator, Henry G. Goldberg, Alex Memory
SAIC
{senatort, goldberghg, memoryac}@saic.com

ABSTRACT
This paper discusses the key role of explanations for applications that discover and detect significant complex rare events. These events are distinguished not necessarily by outliers (i.e., unusual or rare data values), but rather by their *inexplicability* in terms of appropriate real-world behaviors. Outlier detection techniques are typically part of such applications and may provide useful starting points; however, they are far from sufficient for identifying events of interest and discriminating them from similar but uninteresting events to a degree necessary for operational utility. Other techniques that distinguish anomalies from outliers, and then enable anomalies to be classified as relevant or not to the particular detection problem are also necessary. We argue that explanations are the key to the effectiveness of such complex rare event detection applications, and illustrate this point with examples from several real applications.

Categories and Subject Descriptors
H.2.8 [**Database Applicaitons**]: Database applications – *data mining*; I.5.2 [**Pattern Recognition**]: Design Methodlogy – *classifier design and evaluation, feature evaluation and selection*.

General Terms
Security

Keywords
Complex Event Detection, Anomaly Detection, Outlier Detection, Explanation

1. INTRODUCTION
In many real applications, the problem to be solved is the discovery and detection of complex rare events. Examples of such applications include detection of money laundering [9], detection of insider trading [2], and detection of insider threats [5], [10]. These applications typically combine two functions: (1) discovery, which means the identification of previously unrecognized event types, and (2) detection, which means the identification of instances of event types. Complex rare events are characterized by contextual combinations of specific actions, often by multiple agents. They typically occur very infrequently. These events manifest in multiple databases, many of which may or may not be initially – or even eventually – observable or available. Such events may result from multiple agents executing many different types of activities simultaneously, all of which are interleaved in the observed data. Often the observed data are insufficient to distinguish between legitimate actions and actions of interest; additional data are required to make this determination by explaining the observed patterns of activity in the available data.

ODD'13 August 11, 2013 Chicago, IL USA.

2. KEY IDEA
The key idea of this paper is that detection and discovery of complex rare events involves three related but distinct levels of abstraction, and that explanations are the essential mechanism to transform between these levels, because events of interest are characterized not by their rarity but rather by their *inexplicability*.[1]

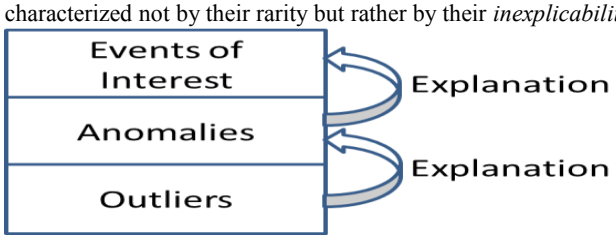


Figure 1: Explanations Enable Transformations Between Outliers, Anomalies and Events of Interest

Such explanations may be provided both implicitly and explicity by the underlying data, models and features incorporated in the system design and by what is presented to a human analyst, respectively.

Explanation provides the ability to infer underlying intent and allows the segmenting of intermingled actions to identify those related to a particular event of interest. This can be as simple as a large number of instances of a condition-action pair, i.e., an anomaly consisting of a set of outliers that would be unlikely to have occurred without intent – from which one can infer the existence of the intent to commit the action when the specified condition occurs. An example of such an anomaly would be repeated personal trading by a stockbroker "ahead" (i.e., in the time period preceding) trades for a customer. The stockbroker would be using his foreknowledge of the customer trade and its likely impact on the price of the security to obtain a profit for himself. A more complex anomaly might be a partial match to a set of observations that would result from execution of a plan to engage in some improper activity. We refer to the behaviors characteristic of such plans as "scenarios" and the corresponding set of observations as "patterns." [8] An example of such a scenario might be an authorized user of a computer system who searches for valuable corporate information in areas distinct from

[1] This idea is related to the concept of "interestingness" in the data mining literature. A specific set of data is "interesting" in these applications because of the real world behavior suggested by its explanation.

his current job responsibilities, downloads such information to removeable media, and then takes an extended vacation.

Three levels of abstraction that we propose and have found useful in various applications we have designed and implemented are: (1) outliers, (2) anomalies, and (3) events of interest, as depicted in figure 1. We define outliers as unusual (combinations of) data points that are statistically rare but may or may not result from the same generative processes as other data points; anomalies as data points that have resulted from a distinct generative process[2] that may or may not appear in the data as outliers; and events of interest as anomalies that are caused by some type of behavior in the real world that is of concern for the application. It is important to note that outliers are always defined with respect to some baseline population and the distribution of values of some (set of) features of entities comprising said population in the database; anomalies are defined with respect to a generative process that results in the observed data; and events of interest are defined by the external needs of an application (e.g., a rule or regulation that may be violated, or a phenomenon of concern for some other reason, such as a combination of symptoms requiring treatment).

For example, in an application to detect insider trading, an outlier might be an unusually large trade, for a particular trader in the context of his or her trading history, in a particular security, on behalf of a particular owner, by a trader compared to other traders during the same timer period, etc. In a public health surveillance application, outliers might be a significant increase in emergency room admissions or in a particular complaint or symptom. In an insider threat detection application, an outlier might be an unusually large number of file transfers to removeable media.

In the insider trading domain, a corresponding anomaly would be a large trade that precedes a material news event and results in a significant profit. It is considered an anomaly because it is generated by a trader reacting to prohibited insider information, not to normal market information. Such an event might be benign – it could simply be a coincidence – or it might be improper, depending on the explanation. An explanation of an improperly caused anomaly could be the connections or communications between the beneficial owner of the profitable trade and a source of the inside information. (A legitimate explanation might be one in which the beneficial owner had recently acquired a large sum of cash and was conducting numerous trades, of which the one in question happened to coincide with an opportunity to make a large immediate profit.) A repeated pattern of such trades would also be an anomaly, providing an analyst with the ability to infer the existence of the unobserved communication of inside information to the trader, such unobserved communication serving as the explanation of the anomaly.

3. EXPLANATIONS

We consider several types of explanations and show examples of how they transform from outliers to anomalies to events of interest. We group explanations into two categories, depending on whether they aid in the transformation from outliers to anomalies or from anomalies to events of interest.

3.1 Explaining Outliers

Because outliers are defined statistically, in terms of where a particular (set of) data points appear on a distribution of values of particular features on a specified population, the explanations of such outliers must depict where the data ponts lie on these distributions with respect to the population. This is not sufficient, however, for several reasons. First, many users are not trained in statistics and will not appreciate an explanation in terms of p-values, t-tests, and the like. Second, and at least as important, the selection of both the underlying population – and of the definition of the entities of interest – typically involves many implicit and explicit assumptions and choices regarding comparison baselines that affect the recognition of outliers. Third, an explanation of the statistical significance and location of an outlier does not tell a user how likely such an outlier is to suggest the existence of the event of interest. Finally, there is the multiple comparison problem – which, while correctable with appropriate statistical methods, can still mislead users of a detection system. We illustrate the second and third of these points with examples, as the first and fourth require no further explanation.

3.1.1 Comparison Baselines

Consider an application that aims to detect significant events in financial time-series data. Such an application might look for trades that are unusual. This can mean many things. Even if we specify the features of interest (e.g., total dollar amount of a trade, frequency of trading, time of day, etc.) – which is not the subject of this paragraph or this paper – we have many other choices. For any aggregate feature, we compare its value over some time period to its value over preceding time periods. For any entity – individual or aggregate – we want to compare the feature values to those of other similar entities, for some definition of "similar." To analyze an individual trade, we may want to compare it to other trades by the same person in the same security, in similar securities, or in other securities. We may want to compare time-based features such as average trading amount per day or week

We have identified three time periods that must be specified for any comparison: (1) the temporal-extent over which a feature is computed for a particular entity; (2) the look-back period over which the same feature is computed in order to establish the distribution of values of that feature; and (3) the granularity of the computations during the look-back period. For example, consider again a financial time series application that uses the price-volatility of a security as a feature. We might look for unusual volatility numbers based on hours or days; we might compute the distribution of historical volatility by looking back over one, three, or six months; and we might use the daily or weekly prices as the basis of this computation.

Similarly, we need to specify the population of entities against which a comparison occurs. We might, for example, compare an individual to his or her own behavior, to people in his or her community (i.e., people to whom he or she is connected according to some type of network structure), or to peers (i.e., people with similar job descriptions and functions).

Once we have made the above choices, we then have to decide whether to compare

Table 1: Outlier Detection Data Structure

ResultScore	ResultMetadata
runID (KEY)	runID
flowID (KEY)	flowID
alfoID (KEY)	algoID
dataTypeID (KEY)	dataTypeID (KEY)
nodeID	entityXtent (KEY)
rawScore [optional]	featureID [optional]
normScore	EntityTemp
rankedScore	popXtent
Rank	popSubXtent
endDate	popTemp
analystScore	scoreMean
hasAnalystScore	scoreStdev
analystUpdateDate	scoreCount
	parameters

[2] This concise description in terms of generative processes was suggested by Andrew Emmott of Oregon State University.

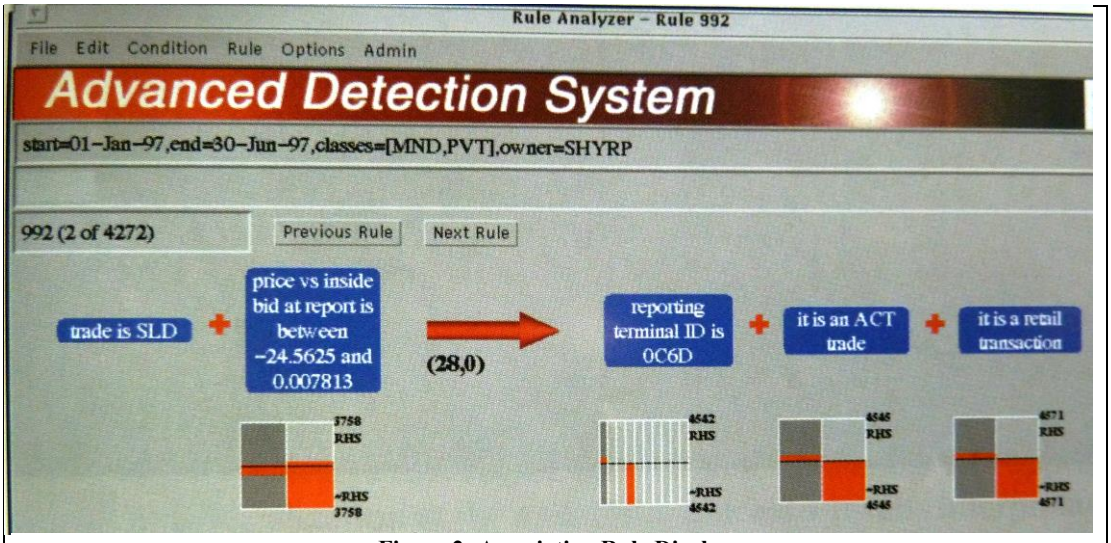


Figure 2: Association Rule Display

absolute values or normalized values. (We may explore a number of possibilities as we design a detection system, to determine which features are the most useful.) If we choose normalized values, we have to determine the basis of normalization. For example, do we normalize the number of communication events of an individual by the number of other individuals with which he has communicated? Do we normalize based on where someone's feature value lies on a distribution based on his/her community and/or peers? Should we normalize based on absolute numbers, units of standard deviation for a particular feature, or percentiles?

However the above choices are made, they need to be communicated explicitly to a human analyst as part of the explanation of an outlier. If multiple versions of such choices are made, then outliers need to be explained in the context of all of these for a full understanding.

3.1.2 How Unusual is an Outlier?

An outlier must be explained not only in terms of its likelihood but also in terms an analyst can understand. Analysts are familiar with their underlying data, so an explanation in these terms has proven effective. It is useful to provide simple visualizations that show where outliers lie according to the distribution of all values of features comprising the outlier. Obviously, such visualizations are limited to two – and occasionally with advanced techniques three – dimensions on a screen, but additional dimensions can be illustrated using color, pattern, iconology, etc. The key idea is that enabling an analyst to see an outlier in the context of the distribution of values of all the relevant features enables the analyst to determine if the outlier is significant. Showing an analyst the number of instances in the data with similar feature values and their resulting interpretation is a useful technique for explaining such outliers.

Figure 2, taken from reference [7] contains an example of such a visualization. This visualization of an association rule, which consists of multiple conditions on both the left and right hand sides, states each condition using pre-stored natural languge text augmented with specific variable values (in the blue boxes in the figure). The bar graphs appearing below the blue boxes depict the number of trades or quotes for which the rule holds or does not, above and below the horizontal axis, respectively, and shows all values of the variable associated with the particular condition across the horizontal axis. The arrow is labeled with the number of instances in the database on the left hand side for which the right hand side does and does not hold. Additional context is provided by the ability to click on the arrow and bring up a table of the raw data summarized in the rule. This visualization enables analysts to understand the context of any prediction made by the rule and evaluate whether the outlier is truly anomalous.

While visualizations are useful for explaining outliers to human

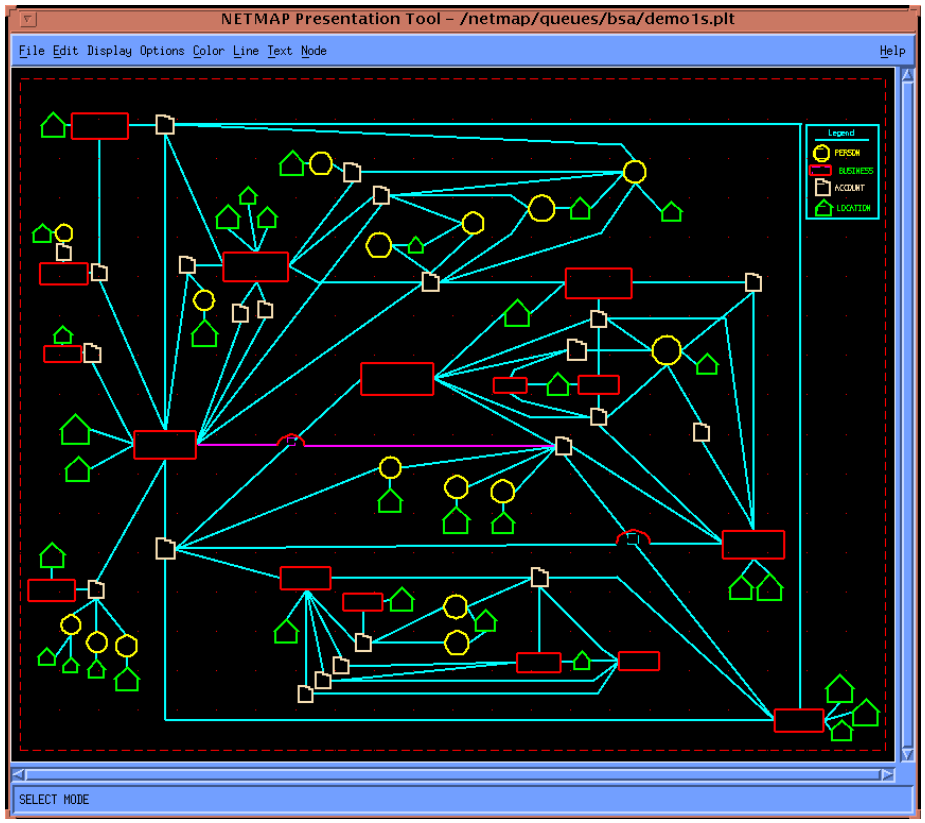


Figure 3: Money Flow Visualization

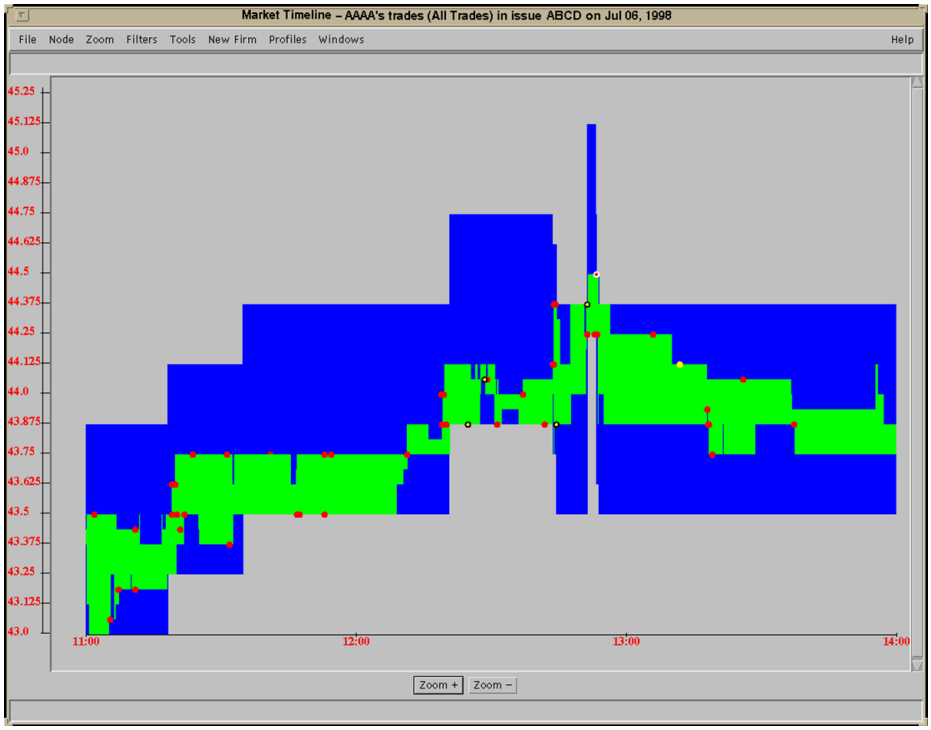


Figure 4: Trade and Quote Visualization

analysts, an automated system requires that detected outliers be stored and available for further analysis. Reference [5] describes a data structure that captures the results of outlier detection, as depicted in table 1. For each run, we allow outlier detection algorithms to compute a raw score, an ordered ranking of all entities, a normalized score, and percentile (ranked) scores. The raw scores may be on different scales and difficult to combine across algorithms. Normalized scores and ranks enable us to compare scores across algorithms. Distributional information such as mean and standard deviation allows us to determine degrees of anomaly.

3.2 Explaining Anomalies

Anomaly explanation differs from outlier explanation by referring to models of processes that may have given rise to the observed data. These models may be based on scenarios derived from domain knowledge or on abstract features of the domain. For example, in the systems described in references [3] and [5], scenarios of behavior are defined and translated to patterns of actions that would be matched in the observed data. The occurrence of matches to these patterns strongly suggests that the behavior modeled by the scenario has occurred. Such scenario-based models may involve multiple interacting agents performing distinct types of actions in a particular temporal sequence or with particular durations.

Other generative models may refer less directly to specific actions in a domain, and capture instead relevant abstractions. For the money laundering detection application discussed in reference [9], the most relevant domain concepts involved money flows between people, businesses and accounts. The visualization depicted in figure 3 was useful to explain a particular anomaly to analysts. In this picture, green "house" shaped icons refer to addresses, yellow circles to people, beige cut-off rectangles to accounts, and red rectangles to businesses. This diagram depicts in the center, three people at three different addresses who share an account which is used by three businesses, which share one other account, and so on. The diagram was used to explain this money-laundering case not only to the analysts management chain, but ultimately to a grand jury. While other aspects of the case, such as the timing and amounts of money flows between the people, accounts, and businesses, is not depicted in the diagram, combined with the accompanying text and briefing, it captured enough information to convince appropriate authorities to proceed. An interesting phenomena that we discovered when developing this visualization is that the explanation that helped an analyst understand the data turned out to be the same visualization that enabled him to explain it to his management chain and to external organizations responsible for further prosecution of the suspected violation.

In the case of the stock market example discussed in reference [3], [7], and [8], the most relevant domain abstractions had to do with temporal sequences of quotes and trades by different market participants. Figure 4 is an actual screen shot of a visualization that captures these abstractions. The x-axis shows the time and the y-axis the price of a particular security. Trades are depicted by dots; quotes by the market maker of interest by the blue band; and the "inside quote" – i.e., the best available bid and ask prices, by the green band.

Other relevant abstractions may be captured as well. Reference [1] uses typical graph structures found to be characteristics in many domains to indentify situations where such structures appear anomalous. Distributional comparisons, such Benford's law (which captures the empirical distribution of the "first digit" in many real sources of data), may also serve as the basis for explaining anomalies.

Finally, a source of explanations for observed anomalies may be additional data types. In the example of insider trading cited earlier, we described two situations which could result in significant, profitable trading in advance of material news. The same event could have resulted from either of two generative processes: (1) the trader had inside information, or (2) the trader had money to invest or losses to cover and bought or sold the stock without knowledge of the insider information. Additional data is required to infer which is the true explanation: call logs, lists of company officers with inside access, names of friends and family, other trading patterns of the trader in questions, etc. In the example of the insider who copies proprietary data, we noted that the files copied were not related to his normal work area. This is an inference which must be made in order to increase our confidence that the event is anomalous and of interest, not just unusual. We can make it either by looking at his other activities in the recent past, or through reference to additional, supplementary data, such as project assignments.

4. ANOMALY DETECTION LANGUAGE

Specifying the functional flow of outlier and anomaly detectors required for a real application, and capturing the multiplicity of choices for baselines and extents, was facilitated by the development and use of a visual anomaly detection lanauge. [4]

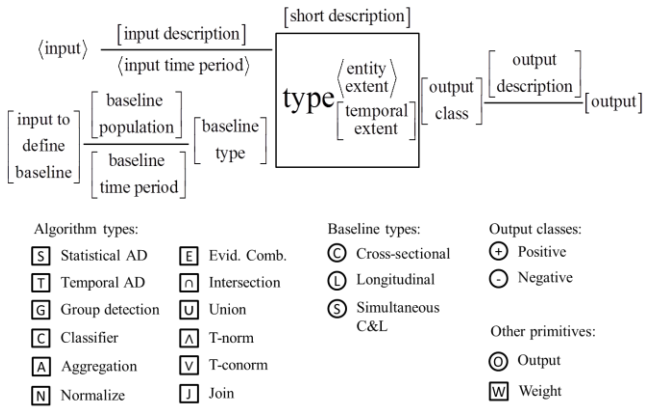


Figure 5: Anomaly Detection Language Syntax

Traditional data flow diagrams cannot express these designs concisely, so we developed a visual anomaly detection language that enables the expression of such combinations of methods, data, baselines, and detection extents. While developed for insider threat detection, the language itself is domain-independent and may be applied to other domains. The language specifies the extent of the entities to be detected (e.g., individual users or groups of users) combined with the temporal extent of potential anomalies. Inputs to these expressions are transactional records of user activity, and outputs are scores on these user-temporal extents.

The syntax of the language is shown in Figure 5; required arguments are in <angle brackets> and optional arguments in [square brackets]. Records are passed along horizontal lines from left to right. Component types are specified by symbols. Entity and temporal extents are super- and sub-scripts, respectively, of component type.

Components may be statistical (denoted by the symbol S) or temporal (T); the latter indicating detectors specialized for anomalies in temporal patterns. Group detectors (G) discover communities of entities, which can be used as baseline populations. Classifiers (C) place input records into classes, which may also be used as baseline populations, or for filtering or partitioning records in general. The classes may be hard, meaning that each record is put into exactly one class, or mixed, in which case a record may be a member of more than one class, possibly to varying degrees. Classifiers might be implemented using a machine-learning method or may be a simple filter based on a lookup on a record. Similarly, aggregators (A) group records with some shared characteristic and summarize their values, e.g., roll-up emails from the same sender to a single record having the count of emails as a new feature; aggregators derive new features from existing ones in this way. Another way to transform features is with a normalizer (N), e.g., rescale real-valued features to the unit interval. Finally, if given a baseline, records are classified and normalized with respect to that baseline.

When sets of records are joined and contain different values for the same feature, ***and*** and ***or*** (∧, ∨) can combine those values, e.g., implement with a t-norm and t-conorm to combine unit-interval values. Evidence combiners (E) also combine values but are more general than ∧ and ∨. And, when no combinations are necessary, union (∪) and intersection (∩) perform the expected operations on input records.

If a baseline is provided, a baseline type specifies how the baseline is used by the component and is indicated by a symbol inside a small circle to the left of the component to which the baseline input connects. With a cross-sectional baseline (C), entity extents are compared to others within the same temporal extent. In contrast, with a longitudinal baseline (L) each entity will be handled individually, and different temporal extents for that entity are compared to one another. A simultaneous baseline (S) combines the first two and compares each input extent to all baseline extents. If a baseline or input time period is not specified, this means that the two cover all available time periods.

Whenever a component may output more than one output class of records, e.g., a binary classifier has (+) and (−) output classes, they should be placed to the right of the component inside circles connected to output lines, unless only one class of output is needed and that class is clear from context, in which case the output class can be omitted.

Weights are scalars in the unit interval used to transform features – usually scores – and are drawn as the letter w inside a rectangle. The type of weighting should be put in a description above the rectangle. Finally, the output of the system is drawn as the letter "O" inside a circle.

Figure 6 uses the anomaly detection language to specify a system for targeting an Intellectual Property (IP) Thief Ambitious

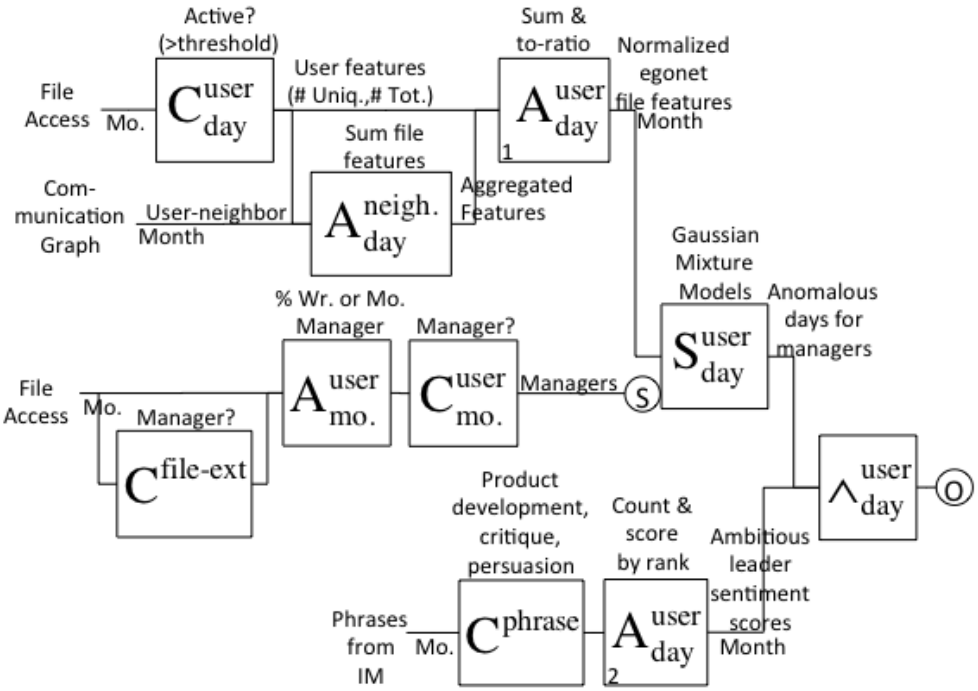


Figure 6: IP Thief – Ambitious Leader Scenario Diagram

Leader scenario in the insider threat detection domain, in which we find a leader of a group of insiders who each steal a few files to be inconspicuous. To counter their strategy, we combine the file activity from the neighbors surrounding each user – known as an egonet – in the IM communication graph, making the leader more anomalous.

We start by filtering user-days to those with sufficient file activity (C_{day}^{user}), then join those records with the IM user-neighbor adjacency list and sum up the features for each "neighbor" $(A_{day}^{neigh.})$. We next add that total for each user to the user's own features and convert the feature totals into ratios $_1(A_{day}^{user})$ that can be compared across egonets of different sizes, e.g. number of unique files to number of all files.

To limit the baseline population to users fitting the profile of a leader, we keep the users $(C_{mo.}^{user})$ with a high fraction of file accesses $(A_{mo.}^{user})$ fitting the manager role according to file extension $(C^{file-ext})$ and use this set as a simultaneous baseline to score (S_{day}^{user}) each user-day.

As an additional indicator, we count $_2(A_{day}^{user})$ phrases seen in IMs between users that fit the scenario (C^{phrase}) and finally combine (Λ_{day}^{user}) with the anomaly scores.

5. COMPLEX EVENT DETECTION SYSTEMS WITH EXPLANATIONS

Real complex event detection systems are multi-layered, consisting of a series of classifiers, each of which is more accurate, primarily because of the availability of additional data to explain outliers or anomalies detected at an earlier stage, and which are biased towards minimizing false negatives at the early stages and minimizing false positives at the later stages [6]. At present, explanations must be provided by humans, sometimes at great cost in terms of investigating false positives or collecting additional data, and requiring significant human analyst involvement and judgement. Adding automated explanation capabilities will enable systems to more accurately distinguish events of interest from other anomalies and enable human analysts to focus their efforts on follow-up investigations and actions for those situations most demanding and worthy of their attention.

Our vision for applications that can be effective at detecting significant complex rare events involves the development of automated explanation techiques to reduce false positive ratios by enabling the transformation of outliers to anomalies and then to events of interest. Techniques that appear promising for such automated explanations might include plan generation, providing the ability to generate plans to accomplish a high-level goal specified in domain terms and translate such plans into an appropriate set of features and baselines that may be present in the data, as well as techniques that could infer potential plans as possible explanations of observed data. Development of ontologies of anomaly types in terms of domain characteristics will be another necessary development to enable this vision.

6. ACKNOWLEDGMENTS

This work contains ideas developed over the course of many years by the authors while employed by several different organizations. It has benefitted from discussions with colleagues in all these organizations. The content of the information in this document does not necessarily reflect the position or the policy of any of these organizations, including the US Government, and no official endorsement by any of these organizations should be inferred.

7. REFERENCES

[1] Chau, D. H., Kittur, A., Hong, J. I. and Faloutsos, C. 2011. Apolo: making sense of large network data by combining rich user interaction and machine learning. In CHI 2011.

[2] Goldberg, Henry G., J. Dale Kirkland, Dennis Lee, Ping Shyr, Dipak Thakker: The NASD Securities Observation, New Analysis and Regulation System (SONAR). IAAI 2003: 11-18

[3] Kirkland, J. Dale, Ted E. Senator, James J. Hayden, Tomasz Dybala, Henry G. Goldberg, Ping Shyr: The NASD Regulation Advanced-Detection System (ADS). AI Magazine 20(1): 55-67 (1999)

[4] Memory, A. et al. 2013. Context-Aware Insider Threat Detection. *Proceedings of the Workshop on Activity Context System Architectures.* Bellevue, WA.

[5] Senator, Ted E., Detecting Insider Threats in a Real Corporate Database of Computer Usage Activity, KDD 2013, to appear.

[6] Senator, T. E. 2005. Multi-stage Classification. In ICDM '05 Proceedings of the Fifth IEEE International Conference on Data Mining Pages 386-393. IEEE Computer Society Washington, DC.

[7] Senator, Ted E. et. al., The NASD Regulation Advanced Detection System: Integrating Data Mining and Visualization for Break Detection in the NASDAQ Stock Market. In *Information Visualization in Data Mining and Knowledge Discovery*, Usama Fayyad, Georges G. Grinstein, and Andreas Weirse, eds., Academic Press (2002)

[8] Senator, Ted E. Ongoing management and application of discovered knowledge in a large regulatory organization: a case study of the use and impact of NASD Regulation's Advanced Detection System (ADS). KDD 2000: 44-53

[9] Senator, Ted E., Henry G. Goldberg, Jerry Wooton, Matthew A. Cottini, A. F. Umar Khan, Christina D. Klinger, Winston M. Llamas, Michael P. Marrone, Raphael W. H. Wong: The Financial Crimes Enforcement Network AI System (FAIS) Identifying Potential Money Laundering from Reports of Large Cash Transactions. AI Magazine 16(4): 21-39 (1995)

[10] Young, W. T et al. 2013. Use of Domain Knowledge to Detect Insider Threats in Computer Activities. Workshop on Research for Insider Threat, *IEEE CS Security and Privacy Workshops*, San Francisco, May 24, 2013.

`

Latent Outlier Detection and the Low Precision Problem

Fei Wang
University of Sydney
Sydney, Australia
fwan7957@it.usyd.edu.au

Sanjay Chawla
University of Sydney and NICTA
Sydney, Australia
sanjay.chawla@sydney.edu.au

Didi Surian
University of Sydney and NICTA
Sydney, Australia
didi.surian@sydney.edu.au

ABSTRACT

The identification of outliers is an intrinsic component of knowledge discovery. However, most outlier detection techniques operate in the observational space, which is often associated with information redundancy and noise. Also, due to the usually high dimensionality of the observational space, the anomalies detected are difficult to comprehend. In this paper we claim that algorithms for discovery of outliers in a latent space will not only lead to more accurate results but potentially provide a natural medium to explain and describe outliers. Specifically, we propose combining Non-Negative Matrix Factorization (NMF) with subspace analysis to discover and interpret outliers. We report on preliminary work towards such an approach.

1. INTRODUCTION

It is well known that new scientific discoveries or "paradigm shifts" are often triggered by the need to explain outliers [11]. The availability of large and ever increasing data sets, across a wide spectrum of domains, provides an opportunity to actively identify outliers with the hope of making new discoveries.

The obvious dilemma in outlier detection is whether the discovered outliers are an artifact of the measurement device or indicative of something more fundamental. Thus the need is not only to design algorithms to identify complex outliers but also provide a framework where they can be described and explained. Sometimes it is easy to explain outliers. For example, we applied the recently introduced k-means-- algorithm [4] on the 2012 season NBA player data set[1]. k-means-- extends the standard kmeans algorithm to simultaneously identify clusters and outliers. The result of the Top-5 outliers are shown in Table 1 and matches with the top players in the NBA "All Star" team. An NBA star is an outlier and given the highly competitive nature of NBA, *an outlier is most likely a star.* Or in other words there are no bad players in the NBA but some players are very good! However, in many other applications it is not at all clear how to proceed to explain outliers. This can be termed as the "Low Precision Problem (LPP)" of outlier detection.

[1]www.basketball-reference.com

ODD'13, August 11th, 2013, Chicago, IL, USA.

Table 1: Given the highly competitive nature of the NBA, not only are stars outliers, but outliers are stars! All the top five outliers are well known leading players of NBA.

Outlier Rank	Player Name	All Star Team (Y/N)
1	Kevin Durant	Y
2	Kobe Bryant	Y
3	LeBron James	Y
4	Kevin Love	N
5	Russell Westbrook	Y

PROBLEM 1. *The Low Precision Problem (LPP) in outlier detection is that*

$$P(\textit{genuine outlier}|\textit{predicted outlier}) \approx \textit{low}$$

LPP occurs because it is hard to disambiguate genuine outliers from errors occurring in the measurement device.

2. THE MULTIPLE SUBSPACE VIEW

A starting point towards addressing LPP and explaining and sifting genuine outliers from measurement errors is to view data from multiple perspectives [12]. In the context where data entities are described by a vector of features, examining an entity in all possible feature subspaces can potentially lead to isolating genuine outliers. This is especially true in high dimensional settings. For example assume that each entity is described by a feature vector of size m. Furthermore, assume that the probability of each feature being recorded incorrectly is p and is independent of other features. Then if m is large, the probability that at least one feature value has been recorded incorrectly is $1-(1-p)^m$ and this can be close to 1 when m is large. Thus having at least one feature value which is corrupted due to measurement error is high. However if we can view the data in multiple subspaces then a genuine outliers will consistently stand out.

A limitation of the multiple subspace approach is that there are exponentially many subspaces leading to intractable algorithms. However the problem can be ameliorated if we notice that in real data sets, the *intrinsic dimensionality* of the data is much lower than the *ambient dimensionality* as we now explain.

3. HIGH-DIMENSIONAL ANOMALIES

It is now part of the data mining folklore that in real data sets, the "degrees of freedom" which actually generate the data is small, albeit unknown. This can be illustrated using examples from computer vision. For example, consider a subset of the Yale Face data shown in Figure 1. Each image is very high-dimensional ($64 \times 64 = 4,096$), however the set of images together live on a three dimensional manifold where the degree of freedom are governed by the rotation of the camera and the lighting. The bottom right hand image (transpose of the top left image) is an outlier as it lives outside the manifold [5].

Thus given a high-dimensional space, if we can project data into a lower-dimension space which preserves the intrinsic structure of the data, then not only can we identify outliers efficiently but potentially explain the discovered outliers. An example of manifold-preserving projection are the family of random projections which preserve pairwise distances with high probability [5]. However, while random projections can lead to improvements in efficiency, by their very nature they make it nearly impossible to interpret the outliers. Thus we need a set of projections to which we can also ascribe some meaning. We next describe matrix factorization methods which are projections of data into lower dimensional space where each dimension aggregates a group of correlated features.

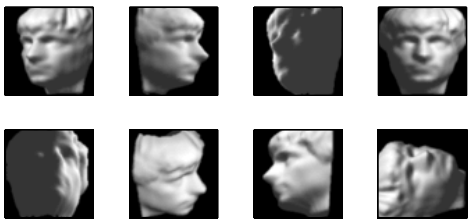


Figure 1: An example to explain the difference between intrinsic and ambient dimension. Samples from the 698-image Yale face data. Each 64 x 64 is a point in a 4,096 dimensional space. However the set of images live in a three dimension set. The bottom right image is added as the transpose of the top left image and is an outlier.

4. MATRIX FACTORIZATION

As we have noted, the challenge in outlier detection is the difficulty to separate true outliers from those data points that are caused because of measurement errors. We have also noted that in high-dimensional space most of the features tend to be correlated. Thus if a data point is a true outlier that fact should be visible in several features. Thus if we take a subspace approach then a genuine outlier will show up as an outlier in more subspaces than an accidental outlier. The challenge in pursuing a subspace approach is that the *space of subspaces* is exponential in the number of features and thus intractable to explore for most practical problems.

One way to address the intractability is to reduce the dimensionality of the original space. This can be carried out using matrix factorization approaches. Factorization is a principled approach of simultaneously aggregating correlated features into a reduced number of "meta-features" which in turn can be imbued with semantics related to the application domain. While Singular Value Decomposition (SVD) and Principal Component Analysis (PCA) have been around for a long time, the recent surge in new methods like Non-Negative Matrix Factorization (NMF) and Bayesian factorization have enhanced the reach of these methods [13]. The key advantage of NMF, say over SVD, is the enhanced interpretation that these methods afford. For example, if X is non-negative document-word matrix or data from a micro-array experiment and $X = UV$ is a non-negative factorization (i.e., both U and V are also non-negative) then the factors can be ascribed a meaning as shown in Table 2.

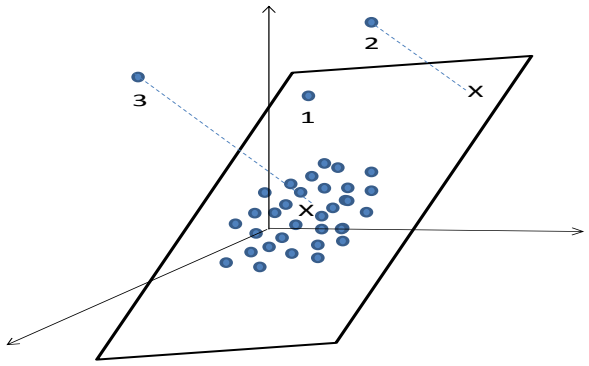


Figure 2: The figure shows the impact of projections of outliers in a lower dimensional space. Data points 1 and 2 remain outliers after projection, while data point 3 is mixed with normal after the projection [8].

4.1 The impact of Projections

Outliers can potentially be impacted in different ways depending upon the nature of outliers. For example, consider the projection shown in Figure 2. The projection shown will have no impact on data point 1, will force data point 3 into a cluster and data point 2 will continue to remain an outlier even though it is far away from the projection plane. Now, which one of these points are genuine outliers is potentially application dependent. However, if we take a subspace perspective, then data point 1 is more likely a genuine outlier. This is because it preserves the correlation between its components but each component is far removed from the main cluster.

4.2 Sensitivity to Outliers

While techniques like NMF provide a promising way to address the combinatorial explosion problem associated with multiple subspace viewing, like SVD, they are highly sensitive to outliers. Thus if our aim is to find outliers, then our method of discovering outliers should not in turn be affected by them. For example, it is well known that both mean and the variance-covariance matrix are extremely sensitive to the presence of even one extreme value and their use for outlier detection will often mask the discovery of genuine outliers. Thus we first have to modify NMF to make them more robust against outliers. Thus we define the following problem:

Table 2: Non-Negative Factorization provides enhanced interpretation of the meta-features. In text processing, the meta-features can be interpreted as topics, while in micro-array analysis, the meta-features are group of correlated genes.

X	U	V
Document-Word	Document-Topic	Topic-Word
Exp-Gene	(Exp,Functional Group)	(Functional Group, Gene)

PROBLEM 2. *[NMF(k,ℓ)] Given a non-negative matrix* $\mathbf{X} \in \mathbb{R}_+^{m\times n}$*, fixed integers* k *and* ℓ*, find matrices* $\mathbf{U} \in \mathbb{R}_+^{m\times k}$*,* $\mathbf{V} \in \mathbb{R}_+^{k\times n}$ *and a subset* $L \subset N$*,* $|L| = \ell$*, which minimizes* $\|X_{-\ell} - UV_{-\ell}\|_F$*, where* $X_{-\ell}$ *is a submatrix consisting of all columns except those from the set* L*.*

To solve the $NMF(k,\ell)$ problem we present the R-NMF algorithm shown in Algorithm 1. The algorithm belong to the class of alternating minimization methods and is very similar to the standard NMF algorithm except for a few caveats. We begin by initializing U in Line 1. In Line 4, we solve for V which minimizes the Frobenius norm of $\|X - U^{i-1}V\|_F$. In Line 5, we compute the residual between X and the current estimate of the product $U^{i-1}V$. In Line 6, we rank the residuals based on the norm of their column values, and L is the index vector of the ranking. We then generate new matrices $X_{-\ell}$ and $V_{-\ell}$ by removing the first ℓ values of the set X and V in Line 7 and 8. In Line 9, we estimate U by minimizing the Frobenius norm of $X_{-\ell}$ and $UV_{-\ell}^i$. We iterate until the convergence criterion is met.

We also propose algorithm O-NMF which is simply using classical NMF algorithm to identify anomaly. The anomalies are calculated by taking ℓ data points which correspond to the top ℓ residual of the final matrices X and UV that is calculated identical to Line 5 of Algorithm 1.

The R-NMF algorithm is an analogous extension of the recently proposed proposed k-means-- algorithm [4]. We should note that another extension for NMF to find outliers has been proposed by Xiong et. al. [14] introduced the method of Direct Robust Matrix Factorization (DMRF). The DMRF method first assumes the existence of a small outlier set S and then infers the low-rank factorization UV by removing S from the data set. It then updates S by using the inferred factorization. In the experiment section we will compare R-NMF with DNMF.

Algorithm 1 [R-NMF Algorithm]

Input: A matrix X of size $m \times n$, m number of features, n number of samples
k the size of the latent space
Output: An $m \times k$ matrix U and $k \times n$ matrix V
$R \approx UV$
1: $U^0 \leftarrow$ random $m \times k$ matrix
2: $i \leftarrow 1$
3: **while** (no convergence achieved) **do**
4: $V^i = \arg\min_V \|X - U^{i-1}V\|_F$
5: $R = X - U^{i-1}V^i$ \\R is a residual matrix
6: Let $L = \{1, 2, \ldots, n\}$ be a new ordering of the columns of R such
$\|R(:,1)\| \geq \|R(:,2)\| \ldots \geq \|R(:,n)\|$
7: $X_{-\ell} \leftarrow X(:, L \setminus L(1:\ell))$
8: $V_{-\ell} \leftarrow V(:, L \setminus L(1:\ell))$
9: $U^i = \arg\min_U \|X_{-\ell} - UV_{-\ell}^i\|$
10: $i \leftarrow i + 1$

The R-NMF algorithm forms the kernel of the subspace algorithm, SR-NMF shown in Algorithm 2 which combines subspace enumeration with R-NMF. Note we only take subspace of the "meta-features." The intuition is that genuine outliers will emerge as outliers in the latent subspaces.

Here we design algorithm that incorporate both the concept of multi subspace view and matrix factorization. As we mentioned before the shortage in [12] is that due to the high dimensionality nature in most of the data set, one simply can not brute force and traversal each and every subspaces. We solve this problem by investigate the problem in a latent space where data are confined in a much small dimensionality.

Algorithm 2 [SR-NMF]

Input: A matrix X of size $m \times n$, m number of features, n number of samples, k the size of the latent space, ℓ number of outliers
Output: A vector R represent the ranking of anomalies with a score in descending order
1: Using $R - NMF$ algorithm we get U and V such that $X \approx UV$
$(U, V) = R - NMF(k,\ell)$
2: $j \leftarrow 0; RANKS \leftarrow$ empty matrix;
3: **STEP1** generate ranks for each subspace
4: **for** $i = 1 \rightarrow k$ **do**
5: generate all set of combinations AS from (k choose i)
6: **for** each $S \in$AS **do**
7: $Residual = X - U(:,S)V(S,:)$
8: $RNorm = columnNorm(Residual)$
9: $[-, RANK] = sort(RNorm, \text{'descend'})$
10: $RANKS = [RANKS; RANK]$
11: $j++$
12: **STEP2** merge ranks into one rank
13: $R \leftarrow$ vector of size n;
14: **for** $i = 1 \rightarrow j$ **do**
15: **for** $p = 1 \rightarrow n$ **do**
16: $R(RANKS(i,p)) = R(RANKS(i,p)) + i$
17: sort R in descending order
$[-, R] = sort(R, \text{'descend'})$ (Note: Matlab Notation)

5. EXPERIMENTS AND RESULTS

In this section we evalute both R-NMF and SR-NMF on several data sets. Our ultimate objective is to verify if SR-NMF can be used to address the **LPP** problem. All our experiments were carried out on a PC with following configurations. Intel(R) Core(TM) i5-2400 CPU @3.1GHz 4GB RAM running on 64-bit Microsoft Windows 7 Enterprise Edition.

5.1 Data Sets

We used three data sets from different application domains which we now describe.

NBA 2012

The NBA 2012 data set consists of nearly two hundred players with each player characterized by twenty features. Example of features include number of points scored, rebounds etc. The data set is freely available from basketball-reference.com.

Spam Email

'Spambase' is a spam email data set [6] consisting of 4,601 emails out of which 1,813 (39%) are spam. The spam e-mails came from their postmaster and individuals who had filed spam and non-spam e-mails from work and personal e-mails. Most of the features (48 out of 57) are frequency of key words.

Research Abstracts

We took around one thousand computer science paper titles from DBLP and also a thousand physics research paper abstracts. We created two data sets. In the first we kept the thousand CS titles and merged them with one hundred physics abstracts. For the second data set, we kept the thousand physics abstracts and merged them with a random subset of one hundred computer science titles. We call the former CSet and the latter PSet.

5.2 Results

We report results on robustness, convergence, runtime and accuracy on the three aforementioned data sets.

Results:Robustness of R-NMF

Here we report on results about the sensitivity of the R-NMF against the classical NMF algorithm used for outlier detection, the O-NMF. We applied both R-NMF and O-NMF algorithm on the NBA 2012 data set but modified one entry in the matrix as a multiple of the mean value. This is shown on the x-axis of Figure 3. For each different value on the x-axis we computed the U matrix and computed the difference in the norm of the new U matrix and the original U matrix. The U matrix is the base matrix and stores the meta-features in terms of the original features.

Figure 3 shows that R-NMF is more robust against perturbations while the U matrix using O-NMF increases without bound. This clearly demonstrates that the traditional NMF algorithm should not be used for any serious applications as it is extremely sensitive to data perturbations.

Results:Convergence Analysis

Here we investigate the convergence properties of the R-NMF algorithms. From Algorithm 1 we know that for each iteration R-NMF will reconstruct U with a given number of outliers excluded. However, each iteration the algorithm may exclude different data points as outliers, this could potentially make the algorithm unstable. Thus, it is necessary to study whether this new algorithm will converge properly.

We conduct the experiments as follows. We use the Spambase data set, and set the number of outliers for R-NMF as the number of spam emails. We vary k and present the results for k=9,12,15, and 18.

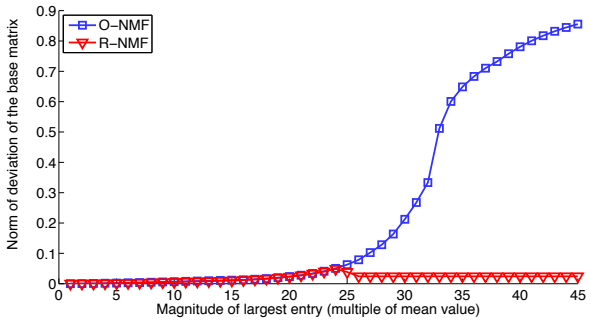


Figure 3: R-NMF is substantially more robust against the presence of outliers in the data compared to standard O-NMF.

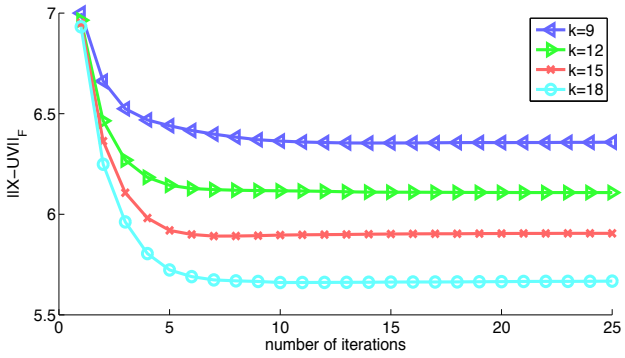


Figure 4: R-NMF converges with all given settings of k. As the dimension of the subspace (k) increases, residual of R-NMF algorithm goes down.

As can be seen from Figure 4, the first thing one can notice is that with bigger k, the residual of the algorithm goes down. This is because with bigger k, the decomposed matrices UV can better reconstruct the original X. Most importantly, the algorithm converge at all given settings of k within 20 repetitions.

Results:Runtime

We present the run time results of R-NMF algorithm for the Spambase data sets in Figure 5 respectively. As expected, we observe that the run time of R-NMF decreases as the number of outliers is increased. This trend follows the intuition of R-NMF algorithm that the construction of base matrix U is based on the data X without the anomalous points (Algorithm 1 line 5-8).

Results:Precision and Recall

We compute precision and recall on the Spambase, PSet and the CSet data sets. The outliers are considered as *positives*. The experiments are conducted as follows. We vary the two variables: k and ℓ, We compared the two proposed algorithms: R-NMF and SR-NMF against the Direct Robust Matrix Factorization (DMRF) approach proposed by [14]. The results for different values of k and different sizes of the outliers specified are show from Table 3-8. At the moment it is hard to draw conclusions from the results. Futher work is

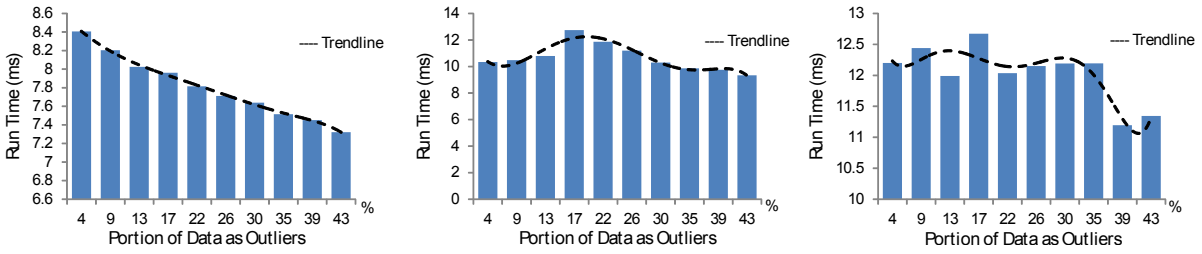


Figure 5: Average Run time R-NMF on Spambase data set: (Left) k = 1, (Middle) k = 2, (Right) k = 3. As the number of outliers increases, the run time for R-NMF decreases. The values here are the average values for all iterations.

Table 3: Precision on CSet: DRMF, SR-NMF and R-NMF.

k	Portion of data as outliers								
	35%			40%			45%		
	DRMF	SR-NMF	R-NMF	DRMF	SR-NMF	R-NMF	DRMF	SR-NMF	R-NMF
6	0.10	**0.13**	**0.13**	0.10	**0.12**	**0.12**	0.09	**0.11**	**0.11**
9	0.09	0.12	**0.14**	0.10	**0.12**	**0.12**	0.09	**0.11**	**0.11**
12	0.10	**0.12**	**0.12**	0.09	0.12	**0.13**	0.09	**0.11**	**0.11**

Table 4: Recall on CSet: DRMF, SR-NMF and R-NMF

k	Portion of data as outliers								
	35%			40%			45%		
	DRMF	SR-NMF	R-NMF	DRMF	SR-NMF	R-NMF	DRMF	SR-NMF	R-NMF
6	0.39	0.49	**0.50**	0.45	0.51	**0.54**	0.47	**0.56**	0.55
9	0.36	0.47	**0.52**	0.45	0.52	**0.54**	0.46	**0.56**	**0.56**
12	0.39	**0.48**	0.47	0.40	0.53	**0.55**	0.45	**0.56**	0.52

Table 5: Precision on PSet: DRMF, SR-NMF and R-NMF.

k	Portion of data as outliers								
	35%			40%			45%		
	DRMF	SR-NMF	R-NMF	DRMF	SR-NMF	R-NMF	DRMF	SR-NMF	R-NMF
6	0.13	**0.15**	0.15	**0.16**	0.15	**0.16**	**0.15**	**0.15**	0.14
9	0.16	0.16	**0.18**	0.16	0.16	0.16	0.15	0.15	0.15
12	0.17	0.16	**0.18**	0.16	0.16	0.16	0.15	0.15	0.15

Table 6: Recall on PSet: DRMF, SR-NMF and R-NMF.

k	Portion of data as outliers								
	35%			40%			45%		
	DRMF	SR-NMF	R-NMF	DRMF	SR-NMF	R-NMF	DRMF	SR-NMF	R-NMF
6	0.49	0.56	**0.58**	**0.72**	0.66	0.70	**0.72**	0.72	0.70
9	0.60	0.60	**0.69**	0.70	0.69	0.70	0.72	**0.73**	0.72
12	0.65	0.60	**0.69**	0.70	**0.72**	0.70	0.72	0.72	**0.73**

Table 7: Precision on Spambase: DRMF, SR-NMF and R-NMF. Best values are highlighted.

k	Portion of data as outliers								
	7%			10%			13%		
	DRMF	SR-NMF	R-NMF	DRMF	SR-NMF	R-NMF	DRMF	SR-NMF	R-NMF
6	0.27	**0.30**	0.29	**0.32**	0.26	0.29	**0.37**	0.32	0.36
9	0.26	0.26	**0.30**	0.28	**0.31**	0.28	0.31	**0.35**	**0.35**
12	0.25	**0.32**	0.30	0.30	**0.33**	0.29	0.30	0.32	**0.36**

required to analyse the results and determine the root cause of the outliers.

6. SUMMARY AND CONCLUSION

Outlier Detection is a core task in data mining. In fact as the size and complexity of data sets increases the need

Table 8: Recall on Spambase: DRMF, SR-NMF and R-NMF. Best values are highlighted.

k	Portion of data as outliers								
	7%			10%			13%		
	DRMF	SR-NMF	R-NMF	DRMF	SR-NMF	R-NMF	DRMF	SR-NMF	R-NMF
6	0.05	**0.06**	0.05	**0.08**	0.07	0.07	**0.12**	0.10	**0.12**
9	0.05	0.05	**0.06**	0.07	**0.08**	0.07	0.10	**0.12**	**0.12**
12	0.04	**0.06**	0.05	**0.08**	**0.08**	0.07	0.10	0.10	**0.12**

to identify meaningful and genuine outliers will only grow. Almost all major applications ranging from health analytic to network data management to bio-informatics require analytical tools which can identify and explain genuine outliers.

The core challenge in outlier detection is to distinguish between genuine and noise outliers. The former are indicative of a new, previously unknown process while the latter is often a result of error in the measurement device. The difficulty to distinguish between genuine and noise outliers leads to the Low Precision Problem (LPP). Our claim is that LPP is the fundamental problem in outlier detection and algorithmic approaches to solve LPP are urgently needed.

One approach to distinguish between genuine and noise outliers is to take a multiple subspace viewpoint. A genuine outlier will stand out in multiple subspaces while a noise outlier will be separated from the core data in much fewer subspaces. However the problem in subspace exploration is that current methods are unlikely to scale to high dimensions.

Matrix factorization methods provide a balanced compromise between full subspace exploration in the feature space versus exploration in the meta-feature or latent space. The advantage of working in the latent space is that many of the features are aggregated into a correlated meta-feature. Often these features in the latent space can be imbued with a semantic meaning relevant to the problem domain. For example, in the case of text mining, the features correspond to words while meta-features correspond to topics.

The challenge with matrix factorization methods is that they are highly sensitive to outliers. This can be a serious problem whenever there is a mismatch between the data and the proposed model. One way to ameliorate the problem is to use an alternate minimization approach to estimate both the matrix decomposition and the outlier set. This is the basis of the NMF(k,ℓ) problem and the R-NMF algorithm. Preliminary results show that R-NMF is substantially more robust compared to a standard NMF approach in the presence of data noise. This opens up a promising avenue for further exploration and address the LPP.

7. RELATED WORK

The task of extracting genuine and meaningful outliers has been extensively investigated in Data Mining, Machine Learning, Database Management and Statistics [3, 1]. Much of the focus, so far, has been on designing algorithms for outlier detection. However the trend moving forward seems to be on detection and interpretation.

While the definition of what constitutes an outlier is application dependent, there are two methods which gained fairly wide traction. These are distance-based outlier techniques which are useful for discovering *global* outliers and density-based approaches for *local* outliers [9, 2].

Recently there has been a growing interest in applying matrix factorization in many different areas, *e.g.* [7],[10]. To the best of our knowledge, probably the most closest work to ours is by Xiong *et al.* [14]. Xiong *et al.* have proposed a method called Direct Robust Matrix Factorization (DRMF) which is based on matrix factorization. DRMF is conceptually based on Singular Value Decomposition (SVD) and error thresholding.

The main algorithm proposed in this paper extends the work on k-means-- proposed in *et al.* [4] which unifies clustering and outlier detection. Furthermore we have taken inspiration from a body of work on multiple subspace outlier detection to distinguish between genuine and accidental outliers [12].

8. REFERENCES

[1] V. Barnett and T. Lewis. *Outliers in Statistical Data*. John Wiley and Sons, 1994.

[2] M. Breunig, H. Kriegel, R. Ng, and J. Sander. Lof: Identifying density-based local outliers. In *IEEE International Conference on Data Mining (ICDM)*, 2000.

[3] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3), 2009.

[4] S. Chawla and A. Gionis. kmeans–: A unified approach to clustering and outlier detection. In *SIAM International Conference on Data Mining (SDM SIAM)*, 2013.

[5] T. de Vries, S. Chawla, and M. E. Houle. Density-preserving projections for large-scale local anomaly detection. *Knowledge and Information Systems (KAIS)*, 32(1):25–52, 2012.

[6] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[7] D. M. Hawkins, L. Liu, and S. S. Young. Robust singular value decomposition. *Technical Report, National Institute of Statistical Sciences*, 2001.

[8] M. Hubert, P. Rousseeuw, and Vanden. ROBPCA: a new approach to robust principal component analysis. *Technometrics*, 47:64–79, 2005.

[9] E. Knorr and R. Ng. Algorithms for mining distance-based outliers in large datasets. In *International Conference on Very Large Data Bases (VLDB)*, 1998.

[10] H.-P. Kriegel, P. Krogel, E. Schubert, and A. Zimek. A general framework for increasing the robustness of pca-based correlation clustering algorithm. In *Scientific and Statistical Database Management Conference (SSDBM)*, 2008.

[11] T. Kuhn. *The Structure of Scientific Revolutions*. University of Chicago, 1962.

[12] E. Müller, I. Assent, P. Iglesias, Y. Mülle, and K. Böhm. Outlier ranking via subspace analysis in

multiple views of the data. In *IEEE International Conference on Data Mining*, pages 529–538, 2012.

[13] A. Singh and G. Gordon. A unified view of matrix factorization models. In *Machine Learning and Knowledge Discovery in Databases*, volume 5212 of *Lecture Notes in Computer Science*, pages 358–373. Springer Berlin Heidelberg, 2008.

[14] L. Xiong, X. Chen, and J. G. Schneider. Direct robust matrix factorizatoin for anomaly detection. In *IEEE International Conference on Data Mining (ICDM)*, pages 844–853, 2011.

Invited Talk

Outlier Ensembles

Charu Agarwal
IBM T.J. Watson Research Center
IBM, New York, USA
charu@us.ibm.com

Abstract

Ensemble analysis is a widely used meta-algorithm for many data mining problems such as classification and clustering. Numerous ensemble-based algorithms have been proposed in the literature for these problems. Compared to the clustering and classification problems, ensemble analysis has been studied in a limited way in the outlier detection literature. In some cases, ensemble analysis techniques have been implicitly used by many outlier analysis algorithms, but the approach is often buried deep into the algorithm and not formally recognized as a general-purpose meta-algorithm. This is in spite of the fact that this problem is rather important in the context of outlier analysis. This talk discusses the various methods which are used in the literature for outlier ensembles and the general principles by which such analysis can be made more effective. A discussion is also provided on how outlier ensembles relate to the ensemble-techniques used commonly for other data mining problems.

Bio

Charu Aggarwal is a Research Scientist at the IBM T. J. Watson Research Center in Yorktown Heights, New York. He completed his B.S. from IIT Kanpur in 1993 and his Ph.D. from Massachusetts Institute of Technology in 1996. His research interest during his Ph.D. years was in combinatorial optimization (network flow algorithms), and his thesis advisor was Professor James B. Orlin. He has since worked in the field of performance analysis, databases, and data mining. He has published over 200 papers in refereed conferences and journals, and has applied for or been granted over 80 patents. Because of the commercial value of the above-mentioned patents, he has received several invention achievement awards and has thrice been designated a Master Inventor at IBM. He is a recipient of an IBM Corporate Award (2003) for his work on bio-terrorist threat detection in data streams, a recipient of the IBM Outstanding Innovation Award (2008) for his scientific contributions to privacy technology, and a recipient of an IBM Research Division Award (2008) for his scientific contributions to data stream research. He has served on the program committees of most major database/data mining conferences, and served as program vice-chairs of the SIAM Conference on Data Mining, 2007, the IEEE ICDM Conference, 2007, the WWW Conference 2009, and the IEEE ICDM Conference, 2009. He served as an associate editor of the IEEE Transactions on Knowledge and Data Engineering Journal from 2004 to 2008. He is an associate editor of the ACM TKDD Journal, an action editor of the Data Mining and Knowledge Discovery Journal, an associate editor of the ACM SIGKDD Explorations, and an associate editor of the Knowledge and Information Systems Journal. He is a fellow of the IEEE for "contributions to knowledge discovery and data mining techniques", and a life-member of the ACM.

ODD'13, August 11, 2013, Chicago, IL, USA.
ACM 978-1-4503-2335-2.

Author Index