Comput Math Organ Theory (2007) 13: 407–422
DOI 10.1007/s10588-007-9018-7

Anomaly detection in a mobile communication network

Alec Pawling · Nitesh V. Chawla · Greg Madey

Published online: 27 September 2007

Abstract Mobile communication networks produce massive amounts of data which may be useful in identifying the location of an emergency situation and the area it affects. We propose a one pass clustering algorithm for quickly identifying anomalous data points. We evaluate this algorithm's ability to detect outliers in a data set and describe how such an algorithm may be used as a component of an emergency response management system.

Keywords Anomaly detection · Communication network · Data clustering · Data mining

1 Introduction

Mobile communication networks have recently received attention as viable, pre-existing sensor networks. City officials in Baltimore use cell phone location data to monitor traffic flow, and the state of Missouri is considering a similar state wide program that would make traffic information available to the public (Associated Press 2005). IntelliOne, a company based in Atlanta, GA, recently released

This paper won the best student paper award at the North American Association for Computational and Organizational Science (NAACSOS) Conference 2006, University of Notre Dame, Notre Dame, IN, USA.

A. Pawling (✉) · N.V. Chawla · G. Madey
Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN 46556, USA
e-mail: apawling@cse.nd.edu

N.V. Chawla
e-mail: nchawla@cse.nd.edu

G. Madey
e-mail: gmadey@cse.nd.edu

a system that displays anonymous cell phone location data onto a map so that users can identify congested areas (National Science Foundation 2006). The emergency response community has also gained interest in using existing cell phone networks as a way to distribute warnings to citizens (Sillem and Wiersma 2006; Wood 2005).

The Wireless Phone Emergency Response (WIPER) system, an emergency response management tool currently under development, monitors an existing mobile communication network. Operating under the assumptions that the behavior of the network models the behavior of a population and that anomalous behavior may indicate an emergency situation has developed, the system attempts to quickly detect anomalies in the network. When anomalies occur, WIPER uses a suite of simulations to predict how the situation will unfold. This paper focuses on the problem of identifying anomalous events in streaming cell phone data as part of the WIPER system. See (Madey et al. 2007; Schoenharl et al. 2006a, 2006b) for a complete overview of the WIPER system.

Our goal is to mine the mobile communication network data for events and anomalies to enable a more efficient emergency response system. Emergency response systems are tools that aid emergency response managers in the decision making process. The difficulty of making good decisions is increased by several factors including stress, fatigue, restrictive time constraints.

Another major issue for emergency response managers is the problem of "information overload." Studies have shown a correlation between abundant available data and bad decision making in crisis situations (Smart and Vertinsky 1977). Good emergency response systems provide access to the large amount of available data in such a way that the emergency response manager can use the data effectively to reach good decisions (Belardo et al. 1984; Jennex 2007).

We believe that an anomaly detection system that monitors incoming streaming mobile communication network data and posts alerts for the emergency response manager will be a useful addition to an emergency response system. It will draw the managers attention to information that may be easily missed in a fast moving, stressful situation. The manager can use or ignore that information based on their experience. The goal is to provide valuable information without being too intrusive in the case of false positives.

The nature of the data poses some difficulties in developing an anomaly detection system. First, a large amount of data arrives at a rapid rate. The sheer volume of the data makes it difficult to store it in its entirety, much less operate on it using repeated accesses, which is a typical algorithmic requirement. Therefore, we aim to develop a method that follows the data stream model (Babcock et al. 2002). Intuitively, a data stream is a sequence of data items that arrive at such a rapid rate that it is only feasible to operate on a small portion of the data. As each data item is seen, it must be either incorporated into a summary that requires a small amount of memory or it must be discarded, in which case it cannot be retrieved. The data stream model imposes two algorithmic limitations: each item in the dataset may only be read once in a predefined order, and memory usage must be sub-linear—typically polylogarithmic with respect to the number of data items seen. Our main focus in this paper is the one pass requirement; we present a one pass hybrid clustering algorithm for anomaly detection.

Another difficulty is the fact that the system is dynamic; the way in which people use the services provided by a mobile communication network changes over time. The anomaly detection approach should be sensitive enough to detect anomalies but should not be so sensitive that it flags changes in the underlying system as anomalies. That said, the cost of false positives is far less than the cost of false negatives. The system can handle the detection of a few non-emergency situations, as long it does not happen too often.

In this paper, we present a one-pass hybrid clustering algorithm for detecting anomalies in streaming data. We evaluate clusters produced by the algorithm and its ability to detect outliers. Finally, we discuss how such an algorithm can be used in an emergency response system like the one described above.

2 Related work

There is abundant literature on the anomaly detection problem which describes a variety approaches, including statistical, neural network, and machine learning methods. In this paper, we focus on the statistical approaches, which can be divided into two categories: parametric and non-parametric. Parametric approaches tend to be more efficient, but assume the data conforms to a particular distribution. Non-parametric methods do not assume any particular data distribution; however, they are often less efficient. (Hodge and Austin 2004; Markou and Singh 2003a, 2003b).

Clustering is an appealing non-parametric method because it allows us to capture various classes of "normal" and "abnormal" behavior. This may be quite useful since, in addition to detecting anomalies caused by events that have never been seen before, knowing various types of "abnormality" would allow us to identify interesting events that have already been seen.

The goal of clustering is to group similar data items together. The concept of similarity is often defined by a distance metric; we use Euclidean distance. Good clustering algorithms form clusters such that the distance between intra-cluster points are minimized and the distance between inter-cluster points are maximized. Anomalies are, intuitively, the data items that are far from all other data items. There are three major types of clustering algorithms: partitional, hierarchical, and incremental (Jain et al. 1999).

2.1 Partitional clustering

Partitional clustering divides the data set into some number, often a predefined number, of disjoint subsets. K-means is a classical and simple clustering algorithm that iteratively refines a set of clusters. The initial cluster centroids for the k-means algorithm are k randomly selected data items from the data set. Each example in the data set is assigned to the closest cluster, and the new cluster centroids are computed. This process is repeated until the clusters stabilize, i.e. no point in the data set receives a new cluster assignment (Witten and Frank 2005).

Expectation maximization (EM) clustering is another classical, partitional algorithm. EM is a probability based algorithm that seeks to discover a set of clusters corresponding to a Gaussian mixture model, a set of Gaussian distributions, that describes the data set. The algorithm is initialized with k random Gaussian distributions and iteratively refines these distributions using a two step process. The expectation step computes the probability that the data set is drawn from the current Gaussian mixture—the likelihood. The maximization step reassigns the data items to the cluster which they most likely belong and recomputes the Gaussian mixture. The algorithm halts when the likelihood that the dataset is drawn from the Gaussian mixture increases by less than a user defined threshold.

There are a couple of drawbacks with these approaches. The k-means and EM algorithms are not guaranteed to find an optimal set of clusters, and both algorithms require a priori knowledge of the number of clusters in the data. These issues can be mitigated by running the algorithms multiple times using different initial conditions and varying numbers of clusters. The best set of clusters is used to describe the data (Witten and Frank 2005). Another issue is scalability. These algorithms are inefficient for very large data sets. Spatial data structures may reduce the time required by these algorithms. kd-trees (Bentley 1975) have been used reduce the number of distance calculations required by k-means. Often, a kd-tree can be used to determine cluster memberships for a subset of points with only k distance computations (rather than k computations for each point in the subset) (Pelleg and Moore 1999). Multiresolutional kd-trees have been used to improve the performance of EM clustering. A multiresolutional kd-tree stores hierarchical summary statistics on the data "owned" by the node: the number of points, centroid, covariance matrix, and bounding hyper-rectangle. With these summaries stored for hierarchical subsets of the data set, the computation of EM parameters can be accelerated significantly (Moore 1999).

2.2 Hierarchical clustering

Hierarchical clustering divides data into a nested set of partitions and may be useful for discovering taxonomies in data. Agglomerative algorithms produce hierarchical clusters via a bottom-up approach in which each example is initially a unique cluster and the clusters are iteratively merged with their neighbors. Two common agglomerative clustering algorithms are single link and complete link. These algorithms are graph based: each example becomes a vertex, and edges are added based on the distance between pairs of vertices. A level of the hierarchical cluster is defined by a distance threshold: an edge is added to the graph if and only if two examples are separated by a distance less than the threshold. The connected components and completely connected components are the clusters for the single link and complete link algorithms, respectively. The hierarchy of clusters is formed by iteratively increasing the threshold to produce larger clusters (Jain and Dubes 1998; Jain et al. 1999). Since single and complete link clustering compute the distances between all pairs of examples in the dataset they have greater time complexity than partitional algorithms, however, they produce optimal solutions.

2.3 Incremental clustering

Incremental algorithms consider each example once, immediately deciding either to place it in a existing cluster or to create a new cluster. These algorithms tend to be fast, but are also often order dependent (Jain et al. 1999). The leader algorithm is a simple incremental clustering algorithm in which each cluster is defined by a single data item—the first item assigned to the cluster. For each data example, if the example is within a user specified distance of the defining item of the closest cluster, the example is assigned to that cluster; otherwise, the example becomes the defining example of a new cluster (Hartigan 1975).

Portnoy et al. (2001) use the leader algorithm for intrusion detection (another application of anomaly detection). In order to handle arbitrary distributions, they normalize the data using the z-score, in which the feature values are transformed by

$$v_i' = \frac{v_i - \bar{v_i}}{\sigma_i}. \tag{1}$$

Unfortunately, this requires two passes over the data. Furthermore, the distance threshold is fixed over all clusters, and cannot change as the data stream evolves.

2.4 Clustering algorithms for streaming data

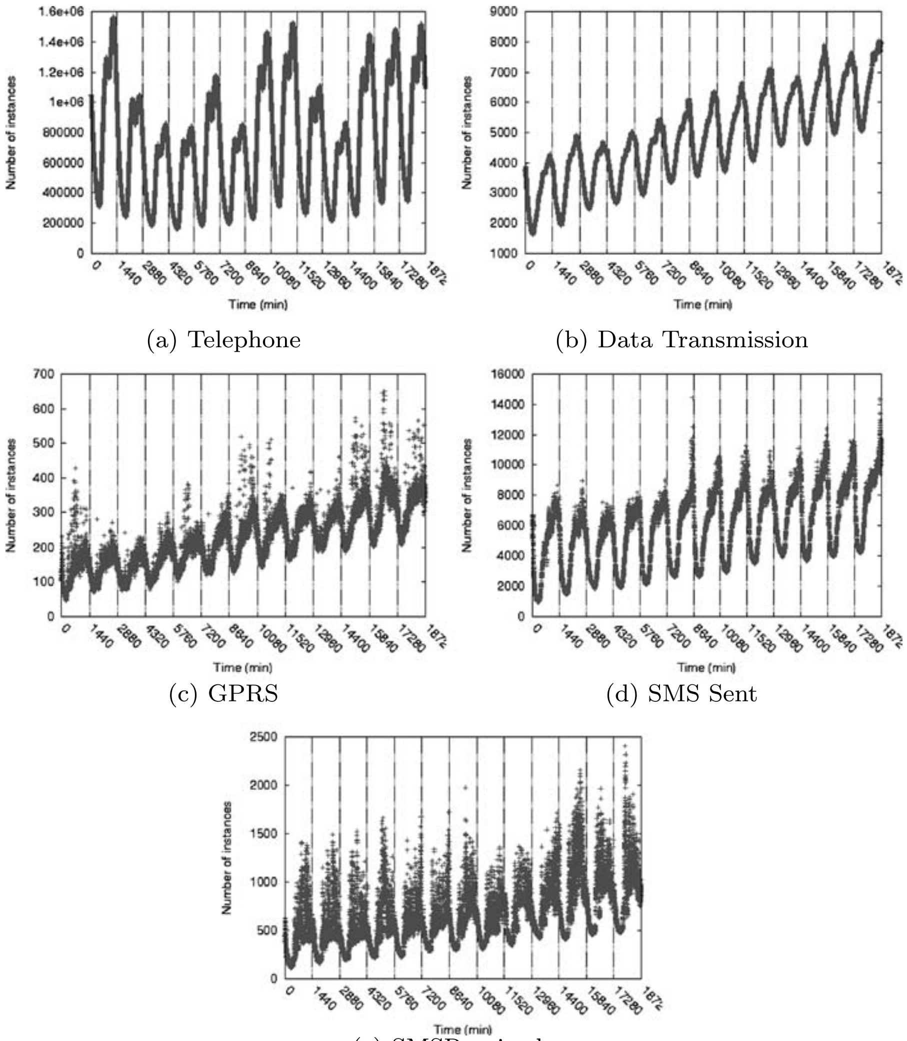
A few methods have been developed for clustering data streams. Guha et al. (2003) present a method based on k-mediods—an algorithm similar to k-means. The clusters are computed periodically as the stream arrives, using a combination of the streaming data and cluster centers from previous iterations to keep memory usage low. Aggarwal et al. (2003) present a method that takes into account the evolution of streaming data, giving more importance to more recent data items rather than letting the clustering results be dominated by a significant amount of outdated data. The algorithm computes *micro-clusters*, which are statistical summaries of the data periodically throughout the stream. These micro-clusters serve as the data points for a modified k-means clustering algorithm.

2.5 Hybrid clustering

Hybrid clustering combines two clustering algorithms. Cheu et al. (2004) examine the use of iterative, partitional algorithms such as k-means, which tend to be fast, as a method of reducing a data set for hierarchical, agglomerative algorithms, such as complete-link, that tend to have high computational complexity. Chipman and Tibshirani (2006) combine agglomerative algorithms—which tend to do well at discovering small clusters—with top-down methods—which tend to do well at discovering large clusters. Surdeanu et al. (2005) propose a hybrid clustering algorithm for document classification that uses hierarchical clustering as a method for determining initial parameters for expectation maximization.

3 The dataset

We use a data set generated from a database of real world mobile communication network information. The database provides the following information for each transaction (use of a service by a customer): the initiation time, the duration (in minutes),



(a) Telephone

(b) Data Transmission

(c) GPRS

(d) SMS Sent

(e) SMSReceived

Fig. 1 Time series for the five service features. These *graphs* show the number of times each service is used during each minute of the 12 day period

and the name of the service. From the database, we generated a data set with 17,280 examples. Each example indicates how many instances of each service are in use for each minute of a 12 day period. We prune the year, month, and day from the data set due to the small time frame covered and we remove 11 services that are rarely used. This leaves a data set with 7 features: hour, minute, data transmission usage, general packet radio service (GPRS) usage, telephone usage, text messages sent, and text messages received.

Figure1 shows the time series for each of the seven service features of the dataset. Note that each time series exhibits periodic concept drift, an underlying change in the process generating the data (Tsymbal 2004), based on the time of day. The telephone time series is relatively consistent from day to day, though the call volume varies somewhat depending on the day of the week and on whether the day is a holiday. In contrast, there is a noticeable increase in the network load for each of the other services as time goes by; this is a form of non-periodic concept drift. This suggests that the way in which people use the telephone service is relatively well established. Notably, this is also the oldest and most used service. As technology evolves, and peoples habits change, we can expect new manifestations of concept drift.

3.1 Offline clustering analysis

Since many clustering algorithms require a priori knowledge of the number of clusters, we must have some way of determining the correct value for this parameter. There are a couple of methods for accomplishing this. One method is to simply perform the clustering for various numbers of clusters and choose the best result based on some metric such as sum squared error or log likelihood. Another method is to use 10-fold cross validation for $k \in 1, 2, \dots, m$, increasing k until the quality of the clustering starts to degrade (Witten and Frank 2005).

We use the implementation of expectation maximization provided by the Weka package (Witten and Frank 2005) with 10-fold cross validation to determine the number of clusters. 10-fold cross validation partitions the data set into 10 equally sized subsets, or folds. Starting with $k = 1$, for each distinct set of 9 folds we compute clusters and the log likelihood of the cluster set. The value of k is incremented by 1 and the process repeats until the average log likelihood is less than that of the previous iteration. The final result is the set of clusters that maximizes the average log

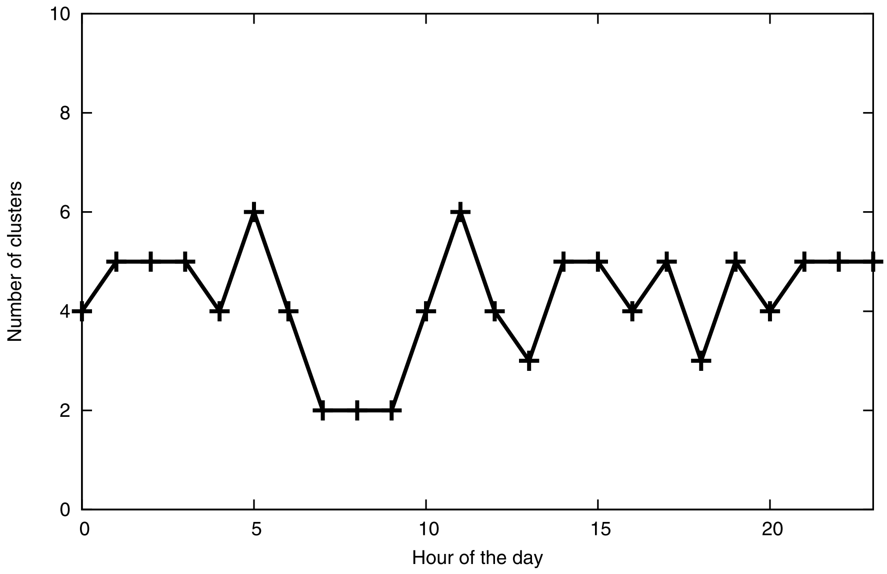


Fig. 2 The number of clusters for each hour of day 4. The number of clusters is determined using 10 fold cross validation with expectation maximization clustering

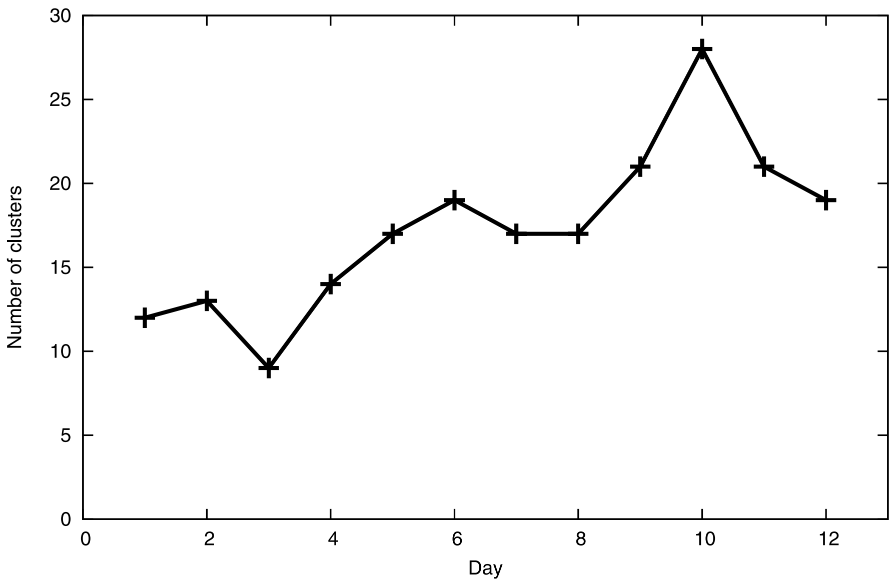


Fig. 3 The number of clusters of the cumulative data set over the 12 days. The number of clusters is determined using 10 fold cross validation with expectation maximization clustering

likelihood. While this approach is not necessarily likely to find a global maxima, it is consistent with Occam's Razor in favoring a smaller number of clusters, which corresponds to a simpler hypothesis (Witten and Frank 2005).

We use expectation maximization to cluster the dataset in the following two ways. First, we arbitrarily select a day from the data set (day 4) and compute the clusters for each hour of the day (see Fig 2). Second, we compute clusters for accumulated data. We cluster the first day, the first two days, the first three days, and so on until we include all 12 days of data (see Fig. 3). Using both approaches, we find that the number of clusters fluctuates, indicating that the appropriate value for k changes as the stream progresses.

4 A hybrid clustering algorithm

We implement a hybrid clustering algorithm that combines a modification of the leader algorithm with k-means clustering. The basic idea behind the algorithm is to use k-means to establish a set of clusters and to use the leader algorithm in conjunction with statistical process control to update the clusters as new data arrives.

Statistical process control (Bicking and Gryna 1979) aims to distinguish between "assignable" and "random" variation. Assignable variations are assumed to have low probability and indicate some anomaly in the underlying process. Random variations, in contrast, are assumed to be quite common and to have little effect on the measurable qualities of the process. These two types of variation may be distinguished based on the difference in some measure on the process output from the mean, μ, of that measure. The threshold is typically some multiple, l, of the standard deviation, σ. Therefore, if the measured output falls in the range $\mu \pm l\sigma$, the variance is considered random; otherwise, it is assignable.

Our algorithm represents the data using two structures: the cluster set and the outlier set. To save space, the cluster set does not store the examples that make up each cluster. Instead, each cluster is summarized by the sum and sum squared values of its feature vectors along with the number of items in the cluster. The outlier set consists of the examples that do not belong to any cluster. We rely on the centroid and the standard deviations of the features to summarize and update the clusters, so clusters are only accepted when they contain some minimum number of examples, m. The algorithm periodically clusters the examples in the outlier set using k-means. Clusters which contain at least m items are reduced to the summary described above and added to the cluster set.

Algorithm 1 shows the details of our approach. The algorithm takes three arguments: the minimum number of elements per cluster, m; the number of clusters to

Algorithm 1 INCREMENTAL HYBRID(X, l, k', m)

```
Let $X$ be a list of examples, $\vec{x}_1, \vec{x}_2, \ldots$
Let $l$ be the threshold multiple
Let $k'$ be the number of clusters to produce in the first level
Let $m$ be the minimum number of items required to accept a cluster

Let $C$ be the set of clusters
Let $U$ be the set of unclustered examples

$C \leftarrow \emptyset$
$U \leftarrow \emptyset$
for all $\vec{x} \in X$ do
   Find the closed cluster, $C_i$
   if dist$(\vec{x}, C_i) < |\vec{\sigma}|$ then
      Add $\vec{c}$ to $C_i$
   end if
   if $|U| = km$ then
      $C' \leftarrow k$-MEANS$(k', U)$
      for all $c' \in C'$ do
         if $c'$ contains more than $m$ examples then
            Add $c'$ to $C$
         else
            Put the items in $c'$ into $U$
         end if
      end for
   end if
end for
```

Algorithm 1 The hybrid clustering algorithm

compute with k-means, k'; and a threshold, l that, when multiplied by the magnitude of the standard deviation vector, defines the boundary between "random" and "assignable" variation. (Note that k' specifies the number of clusters for the first level of the hybrid algorithm, not the final number of clusters produced by the algorithm.) For each example that arrives, we compute the closest cluster. If the example is considered an "assignable" variation, i.e. it is further than $l\sigma$ from the closest cluster center (or the set of clusters is empty), the example is placed in the outlier set. Otherwise, if the example is considered a "random" variation, the example is used to update the summary of the closest cluster. When there are $k'm$ examples in the outlier set, cluster these examples with k-means. The new clusters with at least m examples are added to the cluster set, and all the examples in the remaining clusters return to the outlier set.

This algorithm attempts to take advantage of the fact that the mean is sensitive to outliers. By using means as the components of the cluster center and updating the centers whenever a new example is added to a cluster, we hope to handle a certain amount of concept drift. At the same time, we hope that the use of statistical process control to filter out anomalous data prevents the cluster centers from being affected by outlying points. This algorithm does not require a priori knowledge of the number of clusters (recall that the argument k' is only the number of clusters for the first level cluster), since new clusters will form as necessary.

5 Experimental setup

We evaluate our incremental hybrid clustering algorithms against the expectation maximization clustering algorithm. We use the implementation of expectation maximization provided by the Weka package (Witten and Frank 2005) using 10 fold cross validation to determine the baseline for the number of clusters in the data. For the hybrid algorithm, we use $l = 1, 3$ and $k' = 5, 10, 20, 30$. We evaluate the cluster quality using sum square error. We examine the number of clusters and outliers produced by the hybrid algorithm, and we compare the outlier set produced by the hybrid algorithm to outliers determined by an offline algorithm.

6 Results

Figure 4 shows the number of clusters produced by expectation maximization clustering and our hybrid clustering algorithm. As expected, the number of clusters produced by the hybrid clustering algorithm decreases as the threshold, l, increases. Figure 5 shows the average number of outliers resulting from the application of the hybrid clustering algorithm, with error bars. There are a few factors that cause the number of outliers to fluctuate. Recall that we only accept clusters from k-means if they have some minimum number of members, m. Since we cluster when there are mk' members in the outlier set, increasing k' also increases the number of items used by k-means. If all the clusters are approximately the same size, several clusters with nearly m items may remain in the outlier set, increasing the number of outliers found by the algorithm. In contrast, if most of the examples fall in a few clusters, few examples may remain in the outliers set.

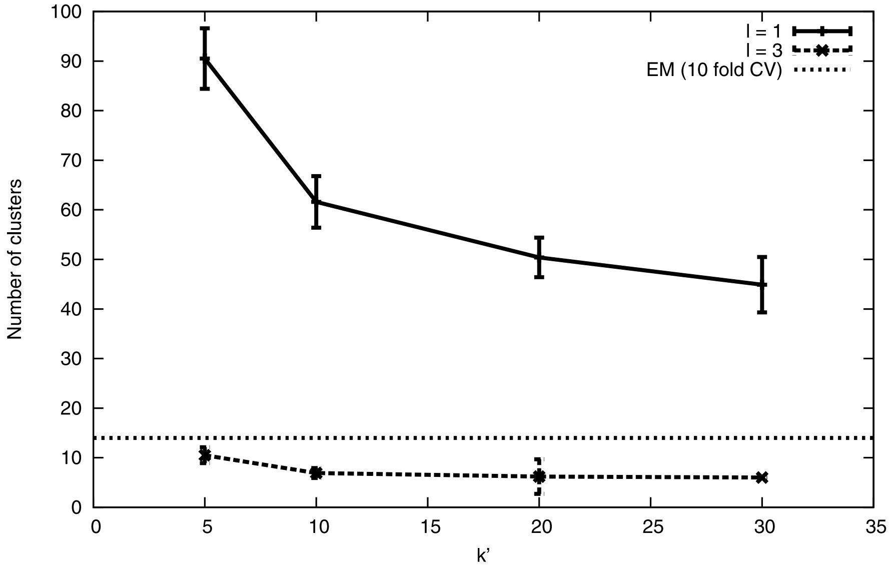


Fig. 4 The mean and standard deviation of the number of clusters produced by the hybrid clustering algorithm. As expected, running the algorithm with a higher threshold value causes it to produce fewer clusters

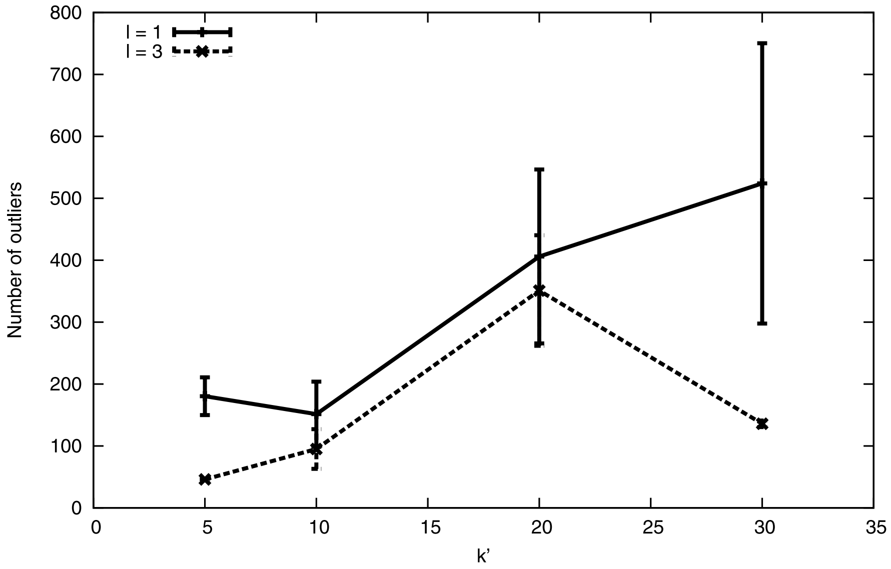


Fig. 5 The mean and standard deviation of the number of outliers resulting from hybrid clustering. As expected, the running the hybrid algorithm with a high threshold value causes it to find fewer outliers

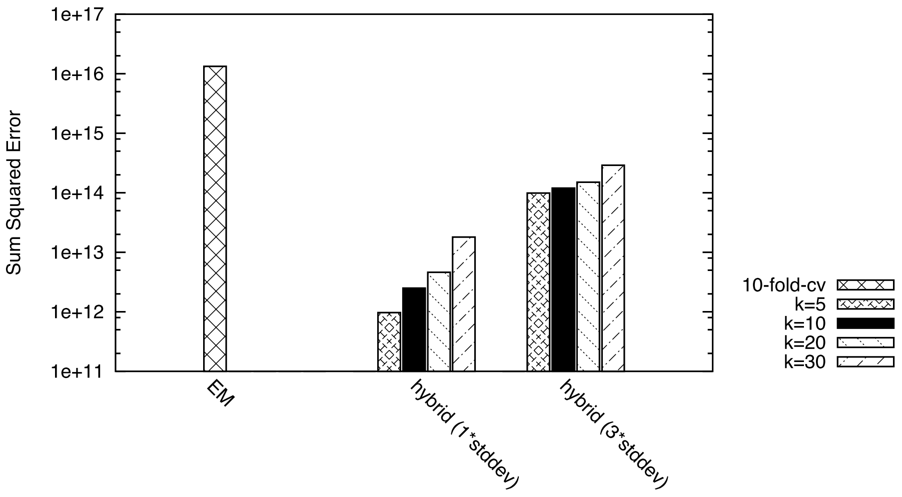


Fig. 6 Sum squared error of the clusters for the expectation maximization and hybrid clustering algorithms. (Note the *y axis* is log-scale.) The hybrid algorithm produces clusters with less sum squared error than expectation maximization

Figure 6 shows the sum squared error for expectation maximization and the hybrid algorithm. The hybrid algorithm produces clusters with less sum squared error, by orders of magnitude, than the expectation maximization algorithm. Also note that the sum squared error increases as both parameters, l and k' increase.

Figures 7 and 8 show the distribution of distances between points in the outlier set and their nearest neighbor in the full data set. Each figure also shows the nearest neighbor distance distribution for the full data set. Recall that we defined outliers as points in the dataset that are far from all other points. We define the extent to which a point is an outlier by its distance from its nearest neighbor. Data points that are closer to their nearest neighbor are less outlying that data points that are far from their nearest neighbor. Ideally, we would like the clustering algorithm to detect all extreme outliers in the nearest neighbor distance distribution for the full data set. These box plots show that this is not the case. The two most outlying examples are never found by the hybrid algorithm, and points below the first quartile in "outlierness" are regularly found. However, for some trials (specifically when $l = 3$ and $k' = 20, 30$), most of the outliers detected by the hybrid algorithm are extreme outliers (see Figs. 8c and 8d).

7 Conclusion

We have discussed issues in anomaly detection on dynamic data streams. We presented a hybrid clustering algorithm that combines k-means clustering, the leader algorithm, and statistical process control. Our results indicate that the quality of the clusters produced by our method are orders of magnitude better than those produced by the expectation maximization algorithm, using sum squared error as an evaluation

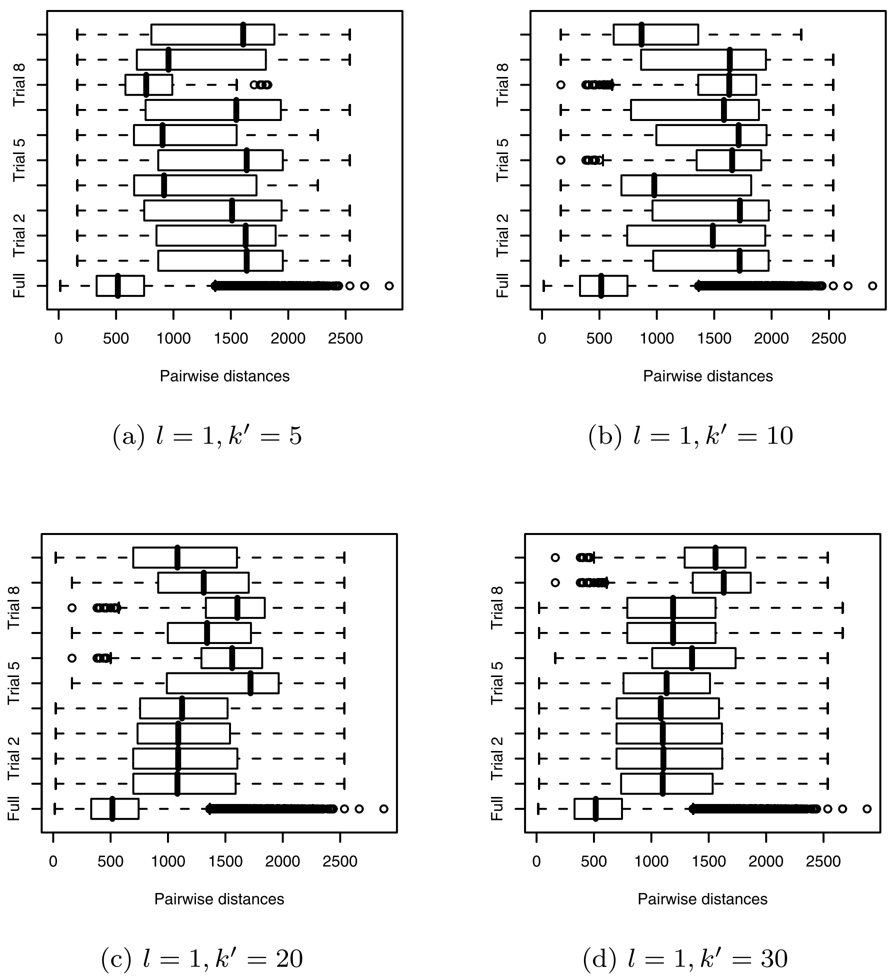
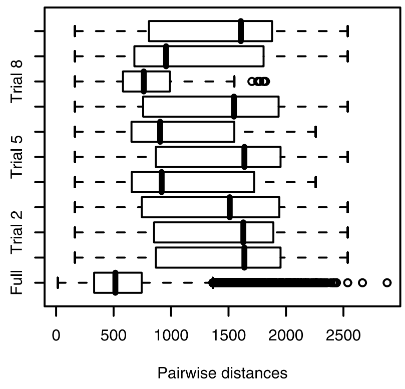


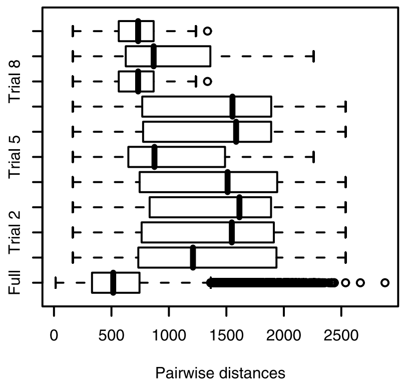
Fig. 7 Box-plots of the distribution of distances between the outliers and their nearest neighbor in the full dataset for each trial of the hybrid clustering algorithm where $l = 1$. The *bottom* box-plot in each graph shows the distribution of distances from the nearest neighbor for all examples in the dataset

metric. We also compared the outlier set discovered by our algorithm with the outliers discovered using one nearest neighbor. While our clustering algorithm produced a number of significant false positive and false negatives, most of the outlier detected by our hybrid algorithm (with proper parameter settings) were in fact outliers. We believe that our approach has promise for clustering and outlier detection on streaming data.

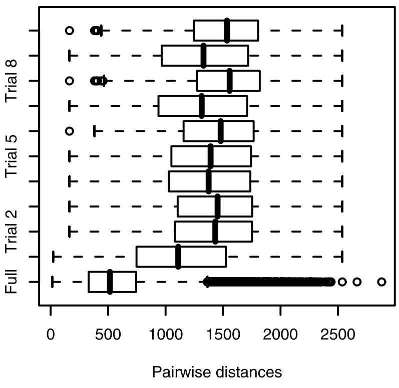
We also believe that this approach has promise for use as a component of the WIPER system. Determining where a new example will be placed—either in an existing cluster or in the outlier set—can be accomplished quickly. It is simply a matter of finding the closest cluster and determining if the example falls within its threshold boundary. Once an initial set of clusters is formed, an alert can be produced whenever



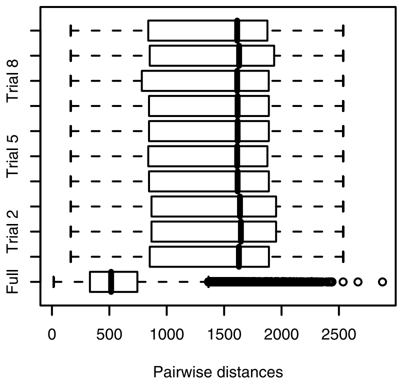
(a) $l = 3, k' = 5$



(b) $l = 3, k' = 10$



(c) $l = 3, k' = 20$



(d) $l = 3, k' = 30$

Fig. 8 Box-plots of the distance between the outliers and their nearest neighbor in the full dataset for each trial of the hybrid clustering algorithm where $l = 1$. The *bottom* box-plot shows the distribution of distances from the nearest neighbor for all examples in the dataset

a data item is assigned to the outlier set. Additionally, data items assigned to a cluster which is known to contain items produced during an emergency situation can also be used to inform the emergency response manager of a potential emergency.

7.1 Future work

We would like to further investigate hybrid clustering algorithms that utilize the leader algorithm and statistical process control. We plan to examine how the clusters change over time as the data stream arrives. We also plan to try different algorithms in place of k-means, particularly an agglomerative algorithm such as complete link clustering.

Acknowledgements The material in this paper is based in part upon work supported by the National Science Foundation, the DDDAS program under Grant No. CNS-050348.

References

Aggarwal CC, Han J, Wang J, Yu PS (2003) framework for clustering evolving data streams. In: Proceedings of the 29th VLDB conference

Associated Press (2005) Tracking cell phones for real-time traffic data

Babcock B, Babu S, Datar M, Motwanim R, Widom J (2002) Models and issues in data stream systems. In: Proceedings of the 21st ACM SIGMOD-SIGACT-SIGART symposium on principles of database systems, pp 1–16

Belardo S, Karwan KR, Wallace WA (1984) Managing the response to disasters using microcomputers. Interfaces 14(2):29–39

Bentley JL (1975) Multidimensional binary search trees used for associative searching. Commun ACM 18(9):509–517

Bicking C, Gryna Jr FM (1979) Quality control handbook. McGraw-Hill, New York

Cheu EY, Keongg C, Zhou Z (2004) On the two-level hybrid clustering algorithm. In: International conference on artificial intelligence in science and technology, pp 138–142

Chipman H, Tibshirani R (2006) Hybrid hierarchical clustering with applications to microarray data. Biostatistics 7(2):286–301

Guha S, Meyerson A, Mishra N, Motwani R, O'Callaghan L (2003) Clustering data streams: theory and practice. IEEE Trans Knowl Data Eng 3:515–528

Hartigan JA (1975) Clustering algorithms. Wiley series in probability and mathematical statistics. Wiley, New York

Hodge VJ, Austin J (2004) A survey of outlier detection methodologies. Artif Intell Rev 22:85–126

Jain AK, Dubes RC (1998) Algorithms for clustering data. Prentice-Hall, Englewood Cliffs

Jain AK, Murty MN, Flynn PJ (1999) Data clustering: a review. ACM Comput Surveys 31(3):264–323

Jennex ME (2007) Modeling emergency response systems. In: Proceedings of the 40th Hawaii international conference on system sciences

Madey GR, Barabási AL, Chawla NV, Gonzalez M, Hachen D, Lantz B, Pawling A, Schoenharl T, Szabó G, Wang P, Yan P (2007) Enhanced situational awareness: application of DDDAS concepts to emergency and disaster management. In: Alexandrov VN, van Albada GD, Sloot PMA, Dongarra J (eds) International conference on computational science. Lecture notes in computer science. Springer, Berlin

Markou M, Singh S (2003a) Novelty detection: a review. part 1: statistical approaches. Signal Process 83(12):2481–2497

Markou M, Singh S (2003b) Novelty detection: a review. part 2: neural network based approaches. Signal Process 83(12):2499–2521

Moore A (1999) Very fast EM-based mixture model clustering using multiresolution kd-trees. In: Kearns M, Cohn D (eds) Advances in neural information processing systems. Kaufman, Los Altos, pp 543–549

National Science Foundation (2006) Real-time traffic routing from the comfort of your car. Press release 06-124. http://www.nsf.gov/news/news_summ.jsp?cntn_id=107972

Pelleg D, Moore A (1999) Accelerating exact k-means algorithms with geometric reasoning. In: Proceedings of the fifth ACM SIGKDD international conference on knowledge discovery and data mining. ACM, New York, pp 277–281

Portnoy L, Eskin E, Stolfo S (2001) Intrusion detection with unlabeled data using clustering. In: ACM workshop on data mining applied to security

Schoenharl T, Bravo R, Madey G (2006a) WIPER: leveraging the cell phone network for emergency response. Int J Intell Control Syst 11(4):209–216

Schoenharl T, Madey G, Szabó G, Barabási AL (2006b) WIPER: multi-agent system for emergency response. In: Proceedings of the 3rd international ISCRAM conference

Sillem S, Wiersma E (2006) Comparing cell broadcast and text messaging for citizen warning. In: Proceedings of the 3rd international ISCRAM conference

Smart C, Vertinsky I (1977) Designs for crisis decision units. Adm Sci Q 22:640–657

Surdeanu M, Turmo J, Ageno A (2005) A hybrid unsupervised approach for document clustering. In: Proceedings of the 5th ACM SIGKDD international conference on knowledge discovery and data mining

Tsymbal A (2004) The problem of concept drift: definitions and related work. Technical report TCD-CS-2004-15, Trinity College, Dublin

Witten IH, Frank E (2005) Data mining: practical machine learning tools and techniques, 2nd edn. Kaufman, Los Altos

Wood M (2005) Cell@lert, for government-to-citizen mass communications in emergencies; it's about time. In: Proceedings of the second international ISCRAM conference

Alec Pawling is a Ph.D. student in the Department of Computer Science and Engineering at the University of Notre Dame. His research interests include machine learning, graph mining, and stream mining.

Nitesh V. Chawla is an Assistant Professor in the Department of Computer Science and Engineering at the University of Notre Dame. His core research in machine learning and data mining focuses on cost/distribution sensitive learning, massively parallel and distributed data mining, semi-supervised learning, and learning in graphs/networks. His recent work has also included various applications of machine learning to systems, bioinformatics, medicine, security, biometrics, and finance. He has served as a co-PI on grants from DoD and DoJ. He has received various awards for his research and teaching. His work with students has resulted in best student paper awards at conferences. He is the recipient of FIE New Faculty Fellowship for his education paper on teaching data mining. He is Associate Editor for the IEEE Transactions on SMC-B. In addition, he has served on organizing and program committees for various workshops/conferences/special issues.

Greg Madey received the Ph.D. and M.S. degrees in operations research from Case Western Reserve University and the M.S. and B.S. degrees in mathematics from Cleveland State University. He is currently an associate professor in the Department of Computer Science and Engineering at the University of Notre Dame. His research includes topics in emergency management systems, web-services and service oriented architectures, bioinformatics, web portals for scientific collaboration, GRID computing, web intelligence, web mining, agent-based modeling and simulation, and swarm intelligence. He has published in various journals, including Communications of the ACM, IEEE Transactions on Engineering Management, IEEE Computing in Science & Engineering, The Journal of Systems & Software, The Journal of MIS, Decision Sciences, The European Journal of OR, Omega, Expert Systems with Applications, and Expert Systems. He is a member of the ACM, AIS, IEEE Computer Society, Informs, and the Society for Computer Simulation.