Dynamic Microcluster Chains in Microtext

Jason R. Robinson and Sherri Condon
The MITRE Corporation, 7525 Colshire Dr., McLean, VA 22012
jrobinson/scondon@mitre.org

Abstract
Two features of microtext that challenge language processing tools are addressed in the context of linking messages in the emergency response domain. First, the effect of very short texts on several classifiers is estimated by comparing the results when classifiers are applied to the full text of news reports vs. only the headlines. These experiments demonstrate a decrease of 5 - 20% in accuracy. A second challenging feature of microtexts is their accumulation in real time, which can be massive for sources such as Twitter. A dynamic hierarchical clustering algorithm is described, and a preliminary experiment in clustering tweets is demonstrated.

1. Introduction
Research on microtexts from social media sources has intensified as applications for analyses have emerged in politics (Diakopoulos, 2010, Mejova and Srinivasan 2012), e-governance (Kwon, Shulman, and Hovy 2006), military communications (Medina 2008, Rosa and Ellen 2009), emergency management (Hughes and Palen 2009, MacEachren et al. 2011), and, with applications like movie ratings, marketing (Claster 2010, Popescu and Etzioni 2005, Stelzener 2010). Researchers and entrepreneurs have quickly produced tools and systems to exploit the information that is available in internet communications.

In the emergency management domain, we are building capabilities to monitor streams of microtexts from both emergency responders and social media like Twitter. We envision a use case in which an emergency operations coordinator could view relevant real-time tweets organized so that tweets about the same or similar events are linked. Users would obtain this information either by submitting queries or by accessing an interface that automatically clusters tweets with similar 911 call records and other responder messages posted to web or chat based emergency communication systems. Therefore, all of the texts that we are working with are microtexts.

At first glance, several features of microtexts present significant challenges to traditional language processing methods. An obvious problem is that text analysis tools operate on language, but microtexts contain very little language. The achievements that researchers have made suggest that language processing tools can operate successfully on microtexts in spite of the minimal content, and it has even been suggested that the small size of microtext messages can be an advantage in some cases (Bermingham and Smeaton 2010). Our systematic comparison provides an estimate of the effect of reducing text size on text analysis tools. This paper compares the performance of standard classifiers when they are applied to the full text of news reports vs. only the headlines.

Another feature of most microtext sources is that content is produced and published rapidly so that a change in production throughput is often indicative of important events. Unlike the typical application of language processing tools to a static corpus, the raw data of social media is constantly changing. With our research focus on analysis of microtexts in the emergency management environment, real time processing is imperative. To address this problem, a dynamic hierarchical clustering algorithm was developed that operates on microtexts as they are passed individually to the clustering routine.

The next section describes related work, and Section 3 presents experiments comparing the results of supervised and unsupervised classifiers operating on microtexts and their analogous full-sized texts. In Section 4, the dynamic clustering algorithm is described, and some evidence of the success of the method is presented. Finally, we provide some conclusions and suggestions for further work.

2. Related Work
Much of the research on microtexts has focused on detection of sentiment and attitudes using a variety of language processing and text mining methods. Among
these, Birmingham and Smeaton (2010) compare classification of binary positive/negative sentiment in Twitter microblogs (tweets) vs. blogs and in Blippr (micro) movie reviews vs. a corpus of archived movie reviews (Pang and Lee 2004). Support Vector Machine (SVM) and Multinomial Naïve Bayes (MNB) classifiers achieved about 5-10% higher accuracies on the microblogs compared to the blogs, but about 1-6% higher accuracy on the movie reviews compared to the micro (Blippr) reviews. These mixed results may be due to many variables in the data that could not be controlled. Our comparison of titles to full text reports uses two datasets with identical topics produced in identical contexts.

Other researchers have experimented with classifying the content of microtexts. Rosa and Ellen (2009) tested 4 classifiers and varying numbers of features to classify US Navy chat, obtaining F1 scores in excess of 0.8 with the k-NN classifier and 400 or more features, all of which are words from the chat messages. In a subsequent study, Rosa et al. (2011) used Twitter hashtags as gold standard identifiers of topic classes for a series of experiments on classifying topics of tweets. Two unsupervised methods (LDA and k-means) performed poorly in the task of clustering tweets that share the same hashtag. A Rocchio classifier (Joachims 1997) achieved an average F1 score of 0.549 for 30 hashtag topics and 0.685 for 6 broader topics (entertainment, fun, sports, money, news, and science). A result that suggests a potentially significant problem for supervised classifiers operating on tweets is a comparison of F1 averages for subsets of the test set that were tweeted during different time intervals. Scores steadily decrease as the tweets in the test set become temporally more distant from the tweets in the training set. This result provides motivation for efforts aimed at improving the performance of unsupervised, dynamic algorithms.

A trend in recent research on microtext content has been to compensate for the lack of information in the text by exploiting other sources of content. Phan, Nguyen and Horiguchi (2008) address the problem of sparseness in short texts such as Google search snippets by augmenting word-based features with topic features. The topics are inferred from topic models estimated by analyzing Wikipedia documents using LDA with Gibbs sampling. Adding the topic features increased accuracy 16-23% to 82.66% when the training set consisted of 6000 snippets.

Ishikawa et al. (2012) also propose to use Wikipedia to weight relationships among words for clustering tweets belonging to hot topics, and Banerjee, Ramanathan, and Gupta, A. (2007) search Wikipedia articles using words from the tweet, then add the titles of the top ranking articles before clustering. In a similar approach, Bernstein et al. (2010) query a web search engine with nouns from the tweet to identify candidate topics. Markman (2011) uses a manually created thesaurus to add words to tweets with less than 3 words after all words except nouns and verbs are removed. Then the tweets are clustered using LDA with Gibbs sampling. Ramage, Dumais and Liebling (2010) use Labeled LDA, which extends LDA by assuming the existence of a set of labels, each characterized by a multinomial distribution. Hong and Davison (2010) extend LDA to an Author-Topic Model in which each word in a document is associated with two latent variables: an author and a topic.

3. Best Case Microtext Classifier Performance

Supervised classifiers are only one type of many language processing tools that researchers currently use, but they are widely used in many applications, often in conjunction with other language processing tools. Also, classifiers do not require extensive customization for specific languages, as tools like parsers and morphological analyzers do.

Classifiers may be an appropriate tool for achieving our goal of linking microtexts in the emergency response domain. The major types of emergencies in which users are interested and the terms people use to describe them change very slowly. Therefore, supervised classification algorithms are good candidates for these functions. Moreover, classifiers are well-understood algorithms that build a static model based on previous data, but once the model has been built, many of these algorithms can classify online/streaming data. These facts motivate a choice of classifiers for comparing the performance of tools on microtexts vs. larger texts.

3.1. ICE Report Data for Classifier Testing

To compare the performance of classifiers on two types of texts, a new dataset for developing and testing topic classifiers was created from news reports that U.S. Immigration and Customs Enforcement (ICE) publishes on the agency's website (http://www.ice.gov/news/). As part of the site's search functionality, the reports are associated with one or more of 17 categories such as Contraband, Document and Benefit Fraud, Gangs, Intellectual Property Rights, Predator, and Human Rights Violators. A corpus of 4,684 reports was compiled by searching on each ICE category and downloading all of the documents that were returned. The resource has been made available at the UCI repository.

In addition to an ICE category, each report in the corpus is associated with a headline. We believe that comparing results using the full text of the reports vs. the headlines provides a best case estimate of classifier performance on microtexts compared to longer texts. Unlike many microtexts, headlines are generally well-formed, and they lack the noisy variation that is common in microtext data. Also, headlines are written to capture key concepts and
assertions in the full report, and therefore, they are likely to be among the optimal representations of the content in the larger text compared to other word sets of equal size.

The average size of the ICE headlines is comparable to the average size of tweets we collected in a corpus of Twitter microblogs (which is described in more detail below). Table 1 presents average sizes of microtexts computed from our data. Words were defined as sequences separated by spaces or punctuation. The averages in Table 1 suggest that the ICE headlines are a suitable example to represent microtexts in our experiments.

### 3.2. Supervised Classification Experiments

In the first of our experiments, we analyzed standard supervised classifier results of microtext compared to the results of the classified analogous full text documents. This experiment provides a baseline for what one should expect while classifying microtexts. Extremely poor performance in initial tests using all of the words motivated textual feature selection techniques to reduce the text to a subset of features: bag of significant words.

#### 3.2.1. Feature Selection

We experimented with two selection techniques. The first is supervised selection, where the most important features per class are selected. The second is unsupervised where features are selected by other criteria such as entropy or cluster relations. We choose Mutual Information as a common supervised selection technique which is very well-understood and Latent Dirichlet Allocation because it has been used extensively to analyze microtexts.

Expected Mutual Information (MI) (Lewis 1992) is an information theoretic test that evaluates whether or not the inclusion of a word makes a given class more likely than the background model. It can be thought of as a ranking function that ranks words higher by how much more different they are than the background model. Here, the background model consisted of words from all ICE categories other than the one under consideration plus a set of New York Times articles.

For the Mutual Information experiments, we selected the 5 top features for each class as well as any other words per class that ranked higher than a specified threshold. The threshold was established by scaling all ranked features to [0,1], where 1 was the highest ranking feature per class and taking all features greater than .5. This technique led to 58 title features and 73 full text features.

Although not necessarily intended for feature selection, Latent Dirichlet Allocation (LDA) (Blei et al. 2003) can be used as an unsupervised feature selection algorithm for documents. LDA assumes that documents are mixtures of a finite set of topics, each of which is a probability distribution of words, and every document has any number of \( n \) topics. LDA models consist of a fixed number of topics, each of which is modeled as a weighted distribution over words and document. A document is modeled as a distribution over a fixed number of topics. There are Dirichlet priors on both the topic distributions over words and the document distributions over topics. Gibbs sampling fits the model parameters to a distribution over topics sampled from a Dirichlet: a per-document multinomial generates a topic. This provides a posterior probability estimate that topic \( \tau \) in document \( d \) generated word \( w \). The end result is the most likely words that would have generated each topic.

The user specifies the number of topics that a collection of documents is about and the number of words that are in each topic. The algorithm then creates clusters of the most likely keywords for each of the topics. The LDA results were generated from MALLET (McCallum 2002). The number of topics was set to 17, the same number of ICE classes. The union of the top 5 words that surpassed a .5 threshold for all topics produced 50 title features and 51 full text features.

#### 3.2.2. Results

The experiments in this section approximate the goal of recommending short texts to an appropriate user. Each ICE category can be viewed as corresponding to a different user or group of users. Using ten folds cross validation, we test how well some baseline algorithms would predict the appropriate category label. We use five typical algorithms from WEKA's machine learning toolkit (Hall et al. 2009): JRip, WEKA's implementation of RIPPER (Cohen 1995); J48, WEKA's implementation of C4.5 (Quinlan 1993); WEKA's implementations of Naïve Bayes (NB) and Bayesian Networks (BN); and WEKA's wrapper of libSVM (Chang & Lin, 2011), all with default values.

Each of these algorithms has advantages and disadvantages; however, all perform fast classification once the model has been trained.

The results in Figure 1 are as expected. The scores with the full reports are generally higher than just the titles. This makes sense because there is more content, so that the most effective features are more likely to be found. Even in the case of the LDA approach to feature selection, where the number of features are practically the same, the full text selection produced features that created more

<table>
<thead>
<tr>
<th>Source</th>
<th>Average Characters</th>
<th>Maximum Characters</th>
<th>Average Words</th>
<th>Maximum Words</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICE Titles</td>
<td>73</td>
<td>284</td>
<td>11</td>
<td>36</td>
</tr>
<tr>
<td>ICE Reports</td>
<td>3,096</td>
<td>15,392</td>
<td>475</td>
<td>2,432</td>
</tr>
<tr>
<td>Tweets</td>
<td>82</td>
<td>140</td>
<td>12</td>
<td>68</td>
</tr>
</tbody>
</table>

*Table 1: Size of ICE Headlines Compared to Other Microtexts*
accurate classifiers. Nevertheless, the classification scores of the ICE report titles were not terrible. In fact, the precision scores are all reasonable. Another expected result is that Mutual Information out-performed LDA as a feature selection strategy. This is expected because MI calculates the scores of each word that best differentiates each class from the other classes. LDA does not look at groups of documents as sets of categories. Instead, LDA models percentages of a fixed number of hypothetical topics that may describe the contents of a given document.

For the classification of the titles, J48 had the highest amounts of standard deviation from the average, .14 for MI features and .20 for LDA. With the full text classification, most of the classifiers had much more deviation from their standard scores. For example, J48’s standard deviation was .21 for MI and .16 for LDA, and Naïve Bayes deviated with .35 for MI features and .14 for LDA.

We used supervised classification to provide a best case, baseline scenario. From this we make the following assumptions: 1) if we had used accurately "labeled" training data for tweets, we would expect accuracy rates no higher than those of the ICE experiments; 2) if these labeled Tweets had much more content, we might expect 10-20 percentage point gains in accuracy; and 3) traditional feature selection techniques such as MI or chi square are probably just as good as any other with these short texts.

Unfortunately, this best case scenario also assumes that we have a discrete set of class labels, which we do not in our use case. Emergency responders want to respond to Fires, Burglary, Chemical Spills, etc. and they also want to be aware of other potential emergencies that may or may not have direct class labels. But at the same time, they do not want to waste their time reading about the latest teen heartthrob or special deals on prescription pharmaceuticals. This creates a hybrid classification problem: (1) a set of known classes, (2) one or more relevant "trending" classes, and (3) a junk class. While interesting in its own right, the problem appears intractable for the magnitude of Twitter data. The algorithm and experiments in the next section represent our initial efforts to address these problems.

4. Dynamic Microclustering Experiments

The emergency application that we are building requires a method for analyzing microtext with the following design considerations: (1) incorporates real-time, online data, (2) identifies what is "trending" but also (3) monitors some predefined classes, and (4) filters out irrelevant microtexts. We began by experimenting with a hybrid supervised method that addresses the problem of clustering a set of microtexts which is constantly increasing in size.

Research on clustering algorithms has focused on static text corpora with efforts directed at increasing the speed of the clustering. The algorithm we have explored is a dynamic or online version of the Nearest Neighbor chain algorithm (Benzécri 1982). In our version, a sliding window of relevant cluster chains are implemented in a fixed-size stack of linked-lists microclusters. This means that a new message is put onto the stack, and then linked to its nearest-neighbor --if there is one-- within the sliding window. As new messages are put onto the stack, old messages are dropped off unless they are linked to other messages in the current sliding window. This creates a manageable set of microcluster chains, upon which a number of further analyses may be made.

Initialization is shown in Figure 2. Upon initialization, several models are built. In our use case, the query model \(Q\) is simply a set of regular expressions that some emergency responders currently use on Twitter queries. In practice, these are very simple \(i.e.:\) fires?|smoke; in our implementation, the expressions were derived from the features selected with MI for a set of 1,000 tweets, manually classified in accordance to 911 emergency responder codes. This model attempts to catch the known-classes part of the problem. The background model \(B\), in this case, consists of a bigram model built from a selection of the English Gigaword corpus\(^1\) combined with a set of previous Tweets. The purpose of this model is to select bigrams from Tweets that occur with some statistical significance. Finally, the dataset model \(D\) is instantiated. This model merely defines structural attributes that are relevant for classification. While this model could be created from standard attribute selection techniques, for our use case it is merely an ad hoc set of attributes we

\(^{1}\text{http://www.ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC2003T05}\)
considered important such as the latitude and longitude of a Tweet, if it exists.

Once these models have been built, the process operates on incoming messages. As a message is presented, the features of the Tweet are selected with respect to these models and normalized. In order to normalize, the feature vector needs to be a fixed length for each new message. Each model contributes to a fixed portion of this vector: (1) each regular expression of model $Q$ corresponds to one feature of the feature vector; (2) the top 5 most significant bigrams per message with respect to model $B^2$ are stored as 5 more features; (3) each value of each attribute in model $D$ is stored as the remaining set of features. As long as the vectors are fixed in this fashion, a fixed threshold for similarity measures can be empirically established.

After initialization, the selected features are clustered so that they are linked to the nearest neighbor, as indicated in Figure 3. If the nearest neighbor falls below a distance threshold and it has a count of keywords above some predefined limit, it is added to a candidates list. Once the fixed window has been traversed, if there are any candidates, the current record is linked to the nearest neighbor. Finally, the current record is added to the fixed stack of microclusters.

4.2. Microcluster Experiment

We experimented with this algorithm on the ICE dataset from Section 3 and on a corpus of Twitter data. The Twitter corpus consisted of 5,009,710 tweets produced from mid-December 2011 through mid-January 2012. We wanted to link tweets to 911 call records, and we have access to records from the Los Angeles metropolitan area. Therefore, we adopted a biased sampling method designed to maximize tweets about a series of arson fires that occurred in Los Angeles from December 29 through January 2, when the arsonist was arrested. Each tweet in the corpus was one of the following: (1) Every 1000th tweet from a GNIP subscription to Twitter which provides approximately 10% of the tweets posted each day or (2) tweets that contained one of a set of fire-related terms or (3) tweets sent or received by a user who identified Los Angeles in the user profile. Tweets in groups (2) and (3) were obtained by using regex from the GNIP.

Our first form of interaction with the microclusters was through interactive Spacetree (Plaisant, et al. 2002) visualizations. The Spacetree in Figure 4 gives a reasonable idea of how the data has been clustered. Linked Count refers to the number of linked messages in the microcluster for a fixed window. The varying sized arrows off of the Linked Count indicate the number of messages that have the same Linked Count. The box stemming from the Linked Count boxes indicates key words in the microclusters. The microcluster itself shows the messages that were clustered together. Figure 4 shows that there are no microclusters larger than 10 linked messages.

Figure 5 illustrates microclusters of tweets about the serial 2011 Los Angeles arson fires that were tweeted on the 30th of December. The clusters illustrate strengths and weaknesses of the approach. While 6 of the 8 tweets in the microclusters are very similar and can be independently verified as reporting arson incidents in a small geographic region, they became associated with the two bodybuilding tweets, from which the counter-intuitive key words “with elite” are extracted. Also, the tweets stating that the fire at Genesse Ave. is not the same fire as the one on Romaine unavoidably link the events for any text analysis algorithm that depends on surface proximity of words without taking into account the semantic relations that are expressed.

Some evidence that the clustering algorithm is performing well is the fact that other clusters of tweets about fires are not included in the arson clusters. Figure 6 illustrates a microcluster of tweets about a fire on a Russian nuclear submarine that happened to occur at the same time as the Los Angeles serial arson fires. Not only does the algorithm distinguish tweets about the two fire

```
create query model Q
create background model B
create dataset model D
initialize empty fixed stack C, rampup window size, minDistance, minCount

for each new record r after rampup has passed:
f, - select( r, B, Q, D )
cluster( C, f,, minDistance, minCount )
```

Figure 2: Initialization and main processing

```
procedure cluster
Input: cluster C, record feature set f,
windowLen
Output: adjusted cluster C
initialize empty stack candidates
initialize empty stack scores

for i=0 to |C|:

d = score( C, f, )
if d < minDistance and c > minCount:
    if empty( scores ) or s < peek( scores ):
        push( scores, s )
        push( candidates, C, )
    if |candidates| > 0:
        link( pop( candidates ), f, )
add( C, f, )
```

Figure 3: Cluster algorithm

---

\(^2\) Messages with fewer than 5 are padded with empty values.

\(^3\) http://gnip.com/

---
**Figure 4: Spacetree of ICE Microclusters**

Medican national sentenced to 12 years in federal prison for firearms smuggling. Border smuggling.

Phoenix man sentenced to 20 years for heroin smuggling. South Texas man sentenced to 2 years for cross-border smuggling.

**Figure 5: Spacetree of Twitter microclusters from 12/30/2011**

- Russian nuclear submarine microcluster from 12/30/2011

**Figure 6: Russian nuclear submarine microcluster from 12/30/2011**

-ID:2938 rajece_raja : fire

Russian nuclear sub fire put out: A fire on a docked Russian nuclear submarine should be fully extinguished with... [URL]

Russian nuclear sub fire put out: A fire on a docked Russian nuclear submarine should be fully extinguished with... [URL]

Russian nuclear sub fire put out: A fire on a docked Russian nuclear submarine should be fully extinguished with... [URL]

Russian nuclear sub fire put out: A fire on a docked Russian nuclear submarine should be fully extinguished with... [URL]

Russian nuclear sub fire put out: A fire on a docked Russian nuclear submarine should be fully extinguished with... [URL]

NIEUWS Report: Russia nuclear sub fire extinguished: Report: Russia nuclear sub fire extinguished

A fire on a dock... [URL]

Report: Russia nuclear sub fire extinguished: A fire on a docked Russian nuclear submarine has been fully put ou... [URL]

Report: Russia nuclear sub fire extinguished [URL]

Russia: Nuclear Sub Fire Finally Out [URL]

Russia Says No Radiation Threat From Nuclear Sub Fire: Russian officials said a fire on a nuclear... [URL] #energy #risk

Russia official said a fire on a nuclear submarine had been contained Friday and ruled out... [URL]

Russia submarine fire 'contained': A huge fire that engulfed a Russian nuclear submarine as it was undergoing... [URL]

Russia submarine fire 'contained': A huge fire that engulfed a Russian nuclear submarine as it was undergoing re... [URL]
events, but also all 13 of the tweets in the nuclear submarine cluster are about that event. Similarly, Figure 7 illustrates a cluster about yet another fire event that was reported on the same day: protestors burned a police car and a U.S. flag in Bahrain. Again, the clustering algorithm distinguished the different fire event and produced a cluster in which every tweet was about the same event.

Figure 4 illustrates another weakness of the approach. The Linked Count boxes show that no clusters were larger than 17 tweets, and the tweet clusters show that multiple clusters about the same event can emerge, in part because of the limit of the window size4. The result is highly unsatisfactory for an end user, which led us to explore a variety of visualizations and interactive techniques. One of the more effective methods we have found is "cluster summarization." For the cluster summarization experiment, we used Kmeans clustering and then represented the Kmeans clusters by the centroid tweet and/or the keywords from the keywords Spacetree box that each cluster had in common. Some centroid tweets were excellent summaries. For example, the centroid tweet of the Russian nuclear submarine fire cluster was Russia fights fire on nuclear submarine: Russia said it had brought a blaze aboard a nuclear submarine under control... [URL] and the centroid tweet of one arson cluster was Hollywood arson spree: Extra firefighters to be on duty overnight: Officials plan to increase the number of deputies... [URL]. The centroid tweet for the Bahrain fire cluster was RT @SitUNBhr: #bahrain #14feb #terror #humanright #unsit @un Bahrain:The terrorists who burned tires and disable traffic!! [URL] ... But the arson tweets were still spread across several clusters, one of which had the centroid Chillin in tha car bumpin some Alias Beatz with 2Mex. this track gonna be fire, which does contain both car and fire. In this case, the top key words make it clear that this is an arson cluster: fire, house fire, rt, lacfd, on romaine, arson, car fire, burned, with elite, and @lacfd (=Los Angeles County Fire Department).

5. Conclusions and Future Work

The experiments we have presented demonstrate the challenges that microtexts present for the task of automated content analysis. The transition from full-sized document processing to microtexts requires a shift in perspective. From the traditional text mining and NLP perspectives, a document represents several possibly interrelated concepts, and determining the significance of the relationships among concepts within the document is typically the most important task. Microtexts, on the other hand, are so brief that there are usually only one or two significant concepts, and if there are more than one, they are virtually always related. This suggests that we now need to look more carefully at concept relationships outside of the documents, which is the approach that many researchers have adopted, as we observed in Section 2.

The microcluster algorithm we propose reflects this shift in perspective; however, this also makes many traditional evaluation metrics far less relevant. First of all, what is considered a cluster is time-dependent because only the linked messages within a time interval are members of the microclusters. Even if one assumes the entire time period of the whole test set, measures such as entropy or cohesion may not necessarily be relevant because of the mixed set of textual features and metadata features.

On one hand, many of our figures show clusters of very similar messages, which would score very high on an entropy type evaluation. Conversely, some clusters with very different text would score poorly, but might be highly relevant. In the use case of an emergency, some people could be tweeting direct information about the event, while others who were at the same location and time could be tweeting about things that appear out of context, unless we know there is an emergency. For example, a message such as "Crazy driver on Sunset!" would be considered irrelevant to tweets about police cars surrounding a bank. But if it can be tied to the same location and time, an investigator might want to track down the tweeter commenting on a possible getaway vehicle. Consequently, we use visualization techniques as a first approximation to verify our technique, conceding that better evaluation metrics will need to be researched in the future.

4 Apparent duplication of messages within clusters is due to truncation of distinguishing content such as URLs, whitespace, additional verbiage, and metadata. It should be easy to reduce the near-duplicates, but it may not be necessary with good tools for viewing and navigating the clusters.
6. Acknowledgements

This research was supported by the Emergency Preparedness and Response investment area of the MITRE Innovation Program. We are grateful to George Wilson for several helpful discussions of this work.

7. References


