From Student Research to Intrusion Detection

N. Paul Schembari
Mathematics Department
East Stroudsburg University of Pennsylvania
East Stroudsburg, PA, 18301
USA
schembari@esu.edu

ABSTRACT: *We describe a multi-year project that began as mostly undergraduate research in data mining applied to computer forensics and has now grown into a prototype for an intrusion detection system. The IDS uses data mining with the Bag of Words methodology, creates a matrix model, and clusters the records using k-means and sparse nonnegative matrix factorization. With no training, these clusters are evaluated to determine if they represent normal system actions or attack vectors. This prototype system has accuracy levels similar to supervised systems on a specific data set. We discuss future plans to make improvements with continued student investigation.*

Keywords: Undergraduate Research, Intrusion Detection, Bag of Words, Nonnegative Matrix Factorization

Received: 27 June 2015, Revised 31 July 2015, Accepted 8 August 2015

1. Introduction

Most faculty agree that there is great benefit in student research. Such projects allow students to expand their knowledge in the discipline, often beyond typical course work. Cybersecurity offers many opportunities for student research because of its rapid development and large number of subjects – from psychology to formal methods in computer science. Furthermore, student research will often enhance faculty investigations. We have found that this synergy between faculty and student research is a great benefit to both parties.

Our project involves *intrusion detection*. Often, one of the most difficult questions for a system administrator is: "*Are we under attack*?" That is, has some person or process used our system in a way it should not be used? For example, a US official recently commented on the security of US corporations stating there are those who have been "*hacked*," and those that *don't know* they have been hacked. To help answer this type of question, administrators often use *intrusion detection systems* (IDS) as part of their Defense in Depth strategies. An IDS helps determine if a system has been attacked.

Our project also involves data mining and computer forensics. The X-Engine, our original system [1] developed by mostly undergraduate students with oversight by faculty and a few graduate students, reads text files and then categorizes them using nonnegative matrix factorization based on the words in the documents. The forensics application involves searching for

evidence in a large email set. The documents read by the system are emails, the emails are categorized, and then an investigator ignores the categories which seem more benign and concentrates on the "*interesting*" categories.

We have now decided to apply this data mining system to intrusion detection. We call our new system *a Bag of Words Intrusion Detection System or BoWIDS*, and we assume that an intelligent human operator collects system data. This could be in the form of system logs, connections requests, etc. BoWIDS will handle this data without any need for pre-processing, so ours is a universal approach. Each item collected (e.g., an IP packet) will be considered a document or record. These records will be delimited in some way. Each delimited part of a record is considered a word. Thus, the record, and the set of all records, is considered a Bag of Words.

Then BoWIDS creates a matrix model of the records, and using an unsupervised learning approach clusters the records. This clustering leads us to label some records as "*normal*" and some as "*possible attacks*." Hence BoWIDS informs the administrator to investigate records which may represent attacks. We will see that the results of testing BoWIDS are similar to other methods using supervised learning. Thus, the strength of BoWIDS is that it can use any delimited data, it requires no training data, and it has accuracy levels similar to methods with training.

Also, we have found that working with students in research, outside of the classroom in a simulated work environment, is very conducive to student learning. We base this conclusion on interviews and surveys completed with the students before graduation. We collect this data as part of our program assessment efforts, and it is based on our Security Engineering Internship course. In this course, students can research by helping design a system, writing code, and analyzing the results. If the research is grant-funded, the students intern on the research project as a part-time job. This experience mimics the students future work environment where the faculty acts as client and supervisor, and the students work in teams to complete deliverables under deadlines. Furthermore, this experiential learning has involved many – the overall efforts, involving this project and others, have spanned more than 10 years with over 50 students involved.

The rest of this paper is organized as follows: In Section 2, we include background information and the work related to our implementation along with the possible learning opportunities for students. In Sections 3 and 4, we describe BoWIDS in detail and examine the results of our tests. Then in Section 5, we discuss planned future investigation along with our ideas for more student research.

2. Background and Related Work

In this section we summarize background information for BoWIDS and review related literature. As will be seen, having students work through this background information has allowed them to greatly expand their knowledge of different areas of computer science and mathematics and relate them to cybersecurity.

2.1 Intrusion Detection

Since the 1980s, intrusion detection has been a popular topic in cybersecurity. If we allow the security requirements of our system to be defined by policy, then we say that a "security event" occurs whenever a policy is violated. Hence, an *intrusion* can be defined as a violation of policy, and something the system administrator should investigate. Thus, an *Intrusion Detection System (IDS)* is part of the information system that automatically informs the administrator to investigate system activity which may be a policy violation.

Intrusion detection is a topic studied by our students in their coursework. Hence, while this information is not new for them, creating a prototype IDS is not included in their regular courses. Also, most of the students' laboratory work deals with signature based intrusion detection (based on known patterns), and so this experience has allowed students to study anomaly detection (unexpected behavior) in more detail.

2.2 Bag of Words Model

Assume we have a collection of documents, called a *corpus*, such as a set of emails. If we model this corpus using BoW, then each email is considered one document, and each document is considered equivalent with its collection of words and the frequency of the words. That is, a document is just considered a "bag of words" no matter the order or meaning. As described by Manning et al [2],

"*…the exact ordering of the terms in a document is ignored... We only retain information on the number of occurrences of each term. Thus, the document 'Mary is quicker than John' is … identical to the document 'John is quicker than Mary.*'"

To perform our analysis, we form a two-dimensional matrix, V, where the rows represent the words in the corpus, and the columns represent the documents. Note that the frequency of common words in a document may not be important. For example, a set of emails from a particular university may have the university name frequently repeated. When this term appears in a document, it is not important. Therefore, the matrix V is formed as follows:

$$v_{j,k} = f_{j,k} * \log\left(N / N_j\right) \quad (1)$$

where

$f_{j,k}$ = the frequency of the word j in document k,

N = the total number of documents,

N_j = the number of documents that contain the word j

This scaling reduces the weights of words that repeat in many documents of the corpus.

This model is called the Term Frequency - Inverse Document Frequency (TF-IDF) matrix. In a corpus where there are many words and each document only contains a few of the words, many of the matrix entries will be 0, and hence we have a sparse matrix.

The BoW methodology is used extensively for text mining and can also be used in other areas such as computer vision [3], [4]. BoWIDS shows that it is possible to use BoW to help in intrusion detection by replacing a corpus of documents with a collection of system data.

In our curriculum, only students who have taken particular electives would be aware of the Bag of Words approach. Hence, studying this information in an applied problem increases the students' computer science knowledge.

2.3 Clustering Methods and Our Previous System

When analyzing a corpus, clustering is the process of grouping together documents which are related, usually with regard to meaning. For example, Berry and Browne [5] clustered emails from the Enron Email Sets. They were able to define meaning from the clusters which they entitled "*Fantasy Football*", "*University of Texas*", etc. That is, one set of documents they mathematically grouped together had different words which all dealt with fantasy football (game, play, season, etc.). For intrusion detection, BoWIDS shows that we can apply the same clustering technique on system data.

One of the early methods for clustering vectors was described by MacQueen [6], called the *k*-means method. Assuming that k clusters are desired,

1. Choose a random set of cluster centers
2. Assign each vector to a cluster based on its nearest center
3. Update each cluster center as the mean of the current cluster vectors
4. Calculate the error between the vectors and the centers

The algorithm is repeated until the errors are minimized. [7]

Another method for clustering was proposed by Lee and Seung [8], [9], called *Nonnegative Matrix Factorization* (NMF), a principal algorithm in BoWIDS. With NMF, vectors are placed in a matrix V. Factorization assumes there are two matrices W and H so that $V = W*H$. Furthermore, by forcing the factor matrices to consist of only nonnegative entries, we can think of the elements of W as the weights which are applied to the columns of H to form additive combinations (not subtractive) and recreate V. Hence, the authors state that NMF allows us to learn the parts of objects that are included in the matrix V. For the sake of brevity, we do not rewrite the algorithms; they are well documented [8], [9].

Pascual-Montano et al [10], describe the need for sparseness in *W*. If we have more 0 terms in *W*, we have a smaller decomposition of *V* into the columns of *H*. Hence, they give an algorithm to force sparseness on *W* called Nonsmooth NMF (nsNMF). We have implemented nsNMF in BoWIDS. Further, Kim and Park [11] state that "*Sparse NMF does not simply provide an alternative to k-means, but rather gives much better and consistent solutions to the clustering problem*." In Section 4, we evaluate experimentation using *k*-means and Sparse NMF on intrusion detection.

Using student programmers, we combined these topics to create the X-Engine [1], a system for evidence collection which applies nsNMF. The X-Engine was created for criminal investigators to search large document sets, but in a manner different from keyword searches. In a keyword search, the search words must be known in advance, but the X-Engine creates its own keywords. Second, because the X-Engine actually separates concepts in the corpus, it can distinguish homonyms while keyword searches cannot.

Hence, the X-Engine outputs a set of topics, and it correlates the associated documents for these topics. In a forensics application, the investigator would choose the topics of interest for further investigation, and leave out the documents corresponding to less interesting topics.

Relating these concepts to student learning, clustering, the *k*-means method, and matrix factorization are not part of the students' regular curriculum. Hence, studying this information increases the students' knowledge in computer science and mathematics. In fact, in the past we held special seminars to help students understand this highly technical information. After these sessions, the students were able to help in the creation of the code of the X-Engine.

2.4 Data Sets for IDS Testing

Of course, new IDS methods require test data. One data set which has extensively been used in IDS research comes from the KDD Cup 1999 competition [12] or KDD99. For example, Chetan and Ashoka [13] created a database architecture for intrusion detection and tested their implementation on KDD99. Also, Lee et al [14] analyzed KDD99 and gave an IDS data mining approach. There are many such publications in IDS literature.

However, as described by Tavallaee et al [15] KDD99 is a highly flawed data set. They describe many redundant records which causes bias towards methods which detect these records. Furthermore, they applied multiple methods from other researchers and achieved very high classification rates. They explain that this is because the researchers use random parts of the KDD99 training set as test sets. Thus they created a new data set for IDS testing, called NSL-KDD. Then, they tested different detection methods on this new data and obtained accuracy rates between 42.29% and 66.16%. Accuracy is defined as follows:

$$\text{(Number of True Positives + Number of False Positives) / (Number of Records)} \qquad (2)$$

Based on the work of Tavallaee et al and others, it seems best to avoid IDS research on KDD99, and hence we test BoWIDS using NSL-KDD.

2.5 Clustering and Intrusion Detection

Many researchers have implemented clustering methods for IDS's using the following process:

1. Convert the suspected data into a set of vectors
2. Cluster the vectors with a chosen method
3. Interpret the clusters to define intrusions and normal data

For example, Therdphapiyanak and Piromsopa [16] use *k*-means clustering on KDD99. Furthermore, they determine that 25 clusters is a good number for this data. Zhang et al [17], also use *k*-means on KDD99 and create a new algorithm based on *k*-means. BoWIDS is different than these methods because of our use of the BoW methodology and because we use no training.

Some researchers have analyzed system calls for intrusion detection and applied a type of BoW approach. For example, Kang et al [18], use *k*-means clustering on system calls, where each call is treated as a word. Hence they use the phrase "*bag of system calls*." This method is related to ours, but it only works on very specific data – system calls – whereas BoWIDS can work

universally on all data types. The authors also give results on various supervised and unsupervised methods. In the unsupervised case they apply *k*-means with only 2 clusters. BoWIDS allows for any number of clusters since more clusters gives better detection, as described above [16].

Wang et al [19] give research using NMF on system calls. They form a matrix by partitioning the data into fixed-length blocks. Then they use the simple frequencies for matrix entries. Furthermore, they use training data to decide on block size and to indicate which of the test data is normal or not. Hence BoWIDS improves on this research by allowing for "*words*" of all sizes (not fixed), by using TF-IDF, and by using no training.

Last, we discuss Step 3 of the Clustering IDS process, where clusters need to be categorized as normal data or intrusions. Petrovic et al [20] use *k*-means to cluster KDD99 and then apply a cluster analysis method. Also, the authors label data as normal or not based on parameters computed in advance, which means they are using training. Finally, they limit their work to 2 clusters, which we have seen is inferior.

Denatious and John [21] use the largest cluster (holding the most records) to represent normal data. This choice is based on the percentage of normal instances in the original data. Also, after clustering, heuristics is used to label clusters as normal or not, and these clusters are used to detect attacks in a separate test dataset, which indicates training. As described below, our method of labeling clusters is based on a combination of the approaches of Petrovic et al and Denatious and John with no training.

Concerning student learning, we can see that allowing students to work on various parts of this project clearly expands their knowledge of computer science, computer security, and mathematics. The specific areas of study include information theory, the Bag of Words approach, and clustering methods. This new project will also allow students to further their IDS understanding.

3. BoWIDS

We now summarize our prototype implementation, BoWIDS. Our goal was to pick the best approaches described in the literature and use the better test data. Furthermore, in Section 5 we describe how improvements can be made using student researchers.

3.1 Data Preparation

NSL-KDD was downloaded from the creators' website [22], and consists of both training and test data. Since it was our intention to use no training, we only discuss the test data called KDDTest+. To avoid confusion with the KDD99 data set, we will call this data NSL-KDD.

NSL-KDD consists of 22544 records, each on a comma-separated line. Hence we consider each line as a document, and each word is defined by the delimiters. An example "document" from NSL-KDD is:

2, tcp, ftp_data, SF, 12983, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0.00, 0.00, 0.00, 0.00, 1.00, 0.00, 0.00, 134, 86, 0.61, 0.04, 0.61, 0.02, 0.00, 0.00, 0.00, 0.00, normal, 21

This example shows how NSL-KDD is an encoded form of tcpdump data, with the addition of the last two words, which in this example indicate that this is normal traffic, with a detection difficulty level of 21. The last two words were added for each document by the creators [14]. Other documents would include the attack type such as "*satan*" instead of "*normal*", and difficulty levels ranging from 1 to 21. These last two words were removed in our tests.

In BoWIDS, we read every entry as a string, so in this example "0" is different than "0.00". Also, treating every entry as a string allows BoWIDS to be universally applicable to many types of delimited data. This includes NSL-KDD data as well as log files, IP packets, etc.

Other considerations for data handling concern the creation of the TF-IDF matrix. One decision is whether or not to use *stop-words*. Stop-words are those which will not be considered in the analysis. For example, in a set of emails on a university server, the name of the university might be considered a stop-word. The X-Engine allowed for stop-words, but we have used no stop-words in BoWIDS. A review of NSL-KDD may indicate a possible stop-word is "0" since it has a high frequency, but this is handled as described below.

The second consideration is whether or not very frequent or infrequent words should be removed from the analysis. For example, we may remove the top 10% and bottom 10% of the words in a corpus. In our X-Engine, these parameters chosen by the user. For BoWIDS, we decided to eliminate the top 10% of the words in each execution. None of the lower percentage words were removed. These were our choices for the prototype and others might be considered. The selection of these choices will make good projects for future student research.

As BoWIDS reads NSL-KDD, it forms the TF-IDF matrix. In this case, there are 22544 documents and 4750 words, forming a matrix with over 107 million entries. Because of the matrix size and the speed of processing, we instead used a sampling technique. We chose 5 random samples with 10% of the entries of mutually exclusive data, forming a partition of the 50% of the original data. We call these samples t1, t2, t3, t4, t5.

In summary, to test BoWIDS we formed words by treating each delimited entry as a string, removed the top 10% of the words, and we ran tests on samples with randomly selected distinct lines from NSL-KDD with 2254 entries in each sample.

3.2 Clustering in BoWIDS

Each sample t1, t2, t3, t4, t5 was processed by BoWIDS, to form a TF-IDF matrix. For example, when evaluating t1, there were 2254 documents and 1154 words. Because of equality in the frequency of some words, 23 words were removed from the analysis. BoWIDS then clustered the 2254 documents using nsNMF. We chose the number of clusters = 2, 5, 10, 15, 20, and 25. Also, for comparison purposes we clustered using *k*-means.

The final step is to evaluate the clusters and label some as normal or not. Because we consider a human operator who must decide which records are normal and which need to be inspected, we label the records as NORMAL or INSPECT. Also, the NSL-KDD data is quite unusual compared to typical system data - the majority of NSL-KDD data represents intrusions. We would expect a typical network or system to have a large percentage of normal data, except when under some types of denial of service attacks. Hence, in using NSL-KDD we applied an unusual approach in labeling INSPECT clusters: since the majority of the documents represent intrusions we label the largest clusters as INSPECT:

- For k clusters ($k > 2$), we label the largest $Floor\ (k / 2) + 1$ clusters as INSPECT
- For the 2 cluster case, we use the largest cluster as the INSPECT cluster

This method labels the majority of documents as INSPECT. For example, when the data is partitioned into 15 clusters, we label the largest 8 clusters as INSPECT.

In an ordinary system, using a NORMAL percentage estimate such as 95%, we would label the largest clusters as NORMAL. The parameter 95% can be estimated by the administrator, or can be determined as it has by other researchers. These were decision made for the prototype and more research is necessary to determine if other choices are better. We hope to use another student team to make this determination.

4. BoWIDS Results

We now examine the results of BoWIDS with nsNMF and compare these executions with *k*-means clustering.

4.1 Increasing the Number of Clusters

First, we determined if increasing the number of clusters gave better accuracy as indicated by other researchers. Table 1 shows our results.

Number of Clusters	**2**	**5**	**10**	**15**	**20**	**25**
***k*-Means (%)**	57.90	57.32	56.92	56.88	57.32	56.74
NMF (%)	49.42	51.86	56.08	52.62	54.66	53.02

Table 1. Accuracy on t1 by Number of Clusters

We see that the trend is for NMF accuracy to increase with the number of clusters, while the k-means accuracy decreases slightly. This data is also illustrated in graphic form in Figure 1.

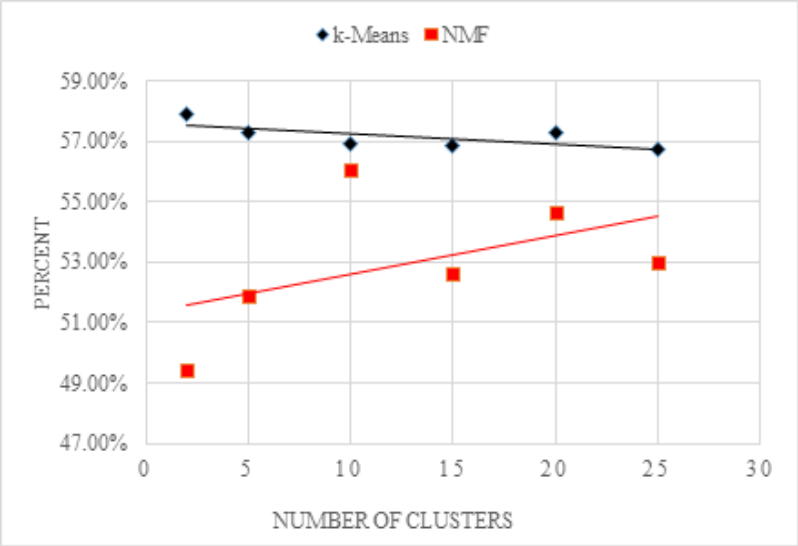

Figure 1. Accuracy by Number of Clusters (t1)

4.2 Comparing NMF and *k*-means

In Figure 1, it is also easy to see that NMF has the best accuracy with 10 clusters. This may be an outlier since the trend is for NMF to improve accuracy with more clusters. Further looking at the graph, we see that the *k*-means accuracy is always greater than the NMF accuracy. This is corroborated by looking at other samples. Table 2 compares *k*-means and NMF in BoWIDS using either 2 or 5 clusters with 5 different samples.

Sample	t1	t2	t3	t4	t5
2 cluster *k*-Means (%)	57.90	55.63	55.63	57.36	57.41
2 cluster NMF (%)	49.42	50.22	54.30	55.19	54.75
5 cluster *k*-Means (%)	57.32	55.32	55.15	56.74	57.36
5 cluster NMF (%)	51.86	55.99	55.41	54.88	54.75

Table 2. Accuracy for Two and Five Clusters

Table 2 shows that NMF accuracy is lower than *k*-means except in two cases.

4.3 *k*-means Results are Flawed

Because it is not possible to show the clusters in this document (some of which are almost 2254 records long), we give the size of the largest clusters. Table 3 shows the size of the largest clusters when using *k*-means on sample t1. The percentage is calculated based on 2254 documents.

Thus when using *k*-means, the algorithm gives one very large cluster, almost the size of the entire sample, and other very small clusters. This largest cluster is the one labeled INSPECT in each case. However, since approximately 40% of NSL-KDD is normal, *k*-means is not a good choice since it tells the administrator to inspect 93% of the documents or more.

Instead with nsNMF, we end up with clusters that are of similar size, and hence give a more equal breakdown of the data. (Recall the comments of Kim and Park [11], who stated that sparse NMF would give better clustering.) Table 4 shows the sizes of the clusters in increasing order using BoWIDS with NMF on sample t1. Thus, when NMF is used, the administrator will be advised

to INSPECT approximately a bit more than 50% of the documents. This is a much more reasonable percentage than the 93% given above for the *k*-Means algorithm. Also, as shown in Figure 1, when a larger number of clusters is used, NMF and *k*-means have similar accuracy levels.

It is important to remember that the NSL-KDD data has a majority of its records as part of attacks – more than 50%. Hence, it is not unreasonable for BoWIDS with NMF to recommend that more than 50% of the documents should be inspected.

Number of Clusters	Largest Cluster Size	% of Documents
2	2245	99.6%
5	2246	99.6%
10	2175	96.5%
15	2202	97.7%
20	2162	95.9%
25	2103	93.3%

Table 3. Largest Cluster Size Using *k*-means for t1

Number of Clusters	Number of Documents Per Cluster
2	1086, 1168
5	382, 427, 432, 454, 559
10	191, 200, 201, 210, 211, 213, 220, 226, 233, 349
15	115, 121, 125, 129, 139, 140, 141, 142, 145, 148, 148, 161, 161, 163, 276
20	83, 91, 93, 94, 98, 100, 103, 103, 105, 105, 108, 108, 109, 114, 117, 119, 119, 121, 122, 242
25	69, 73, 73, 74, 78, 78, 79, 80, 82, 82, 83, 84, 84, 84, 87, 89, 89, 90, 94, 97, 98, 98, 101, 112, 196

Table 4. Cluster Sizes When Using NMF for t1

4.4 BoWIDS Optimal Results

We can also determine the best possible results under BoWIDS with NMF. For this analysis, consider the 25 clusters case. For each cluster, some records are NORMAL and some are INSPECT, even though they have been clustered together. In order to achieve the highest accuracy, a cluster would need to be categorized as NORMAL if it contained a majority of NORMAL records, and a cluster would need to be categorized as INPSECT if it contained a majority of INSPECT records. With this method we obtain 57.90% accuracy as the optimal results. Thus our actual results with accuracies near 55% are near optimal.

4.5 BoWIDS Compared to Other Methods

Finally, we compare BoWIDS results with other methods. The best comparison is with the results of Tavallaee et al [14] since they also used NSL-KDD. Tavallaee's results are shown in Table 5.

Interestingly, the researchers obtained these accuracy levels with training data while BoWIDS uses none. For example, BoWIDS, on sample t1, our results of an average accuracy of 52.94% are very good since we use no training data. Compare this to the Tavallaee result of 42.29% accuracy when analyzing NSL-KDD using Support Vector Machines. Furthermore, if we use BoWIDS with NMF and 25 clusters, our accuracy rose to 53.02% without any training data. Hence our results are quite good, but we still plan for improvements as discussed in Section 5.

5. Future Work

We plan four aspects of new research using student teams. First, we need better data. Testing an IDS on data which consists of

more attacks than normal traffic does not represent a usual system. An ideal situation would be to collect data on our own systems as a test bed. One possibility is to use our university production network - students would create a system for data collection and then anonymize the data. Another possibility is to use our offline lab as a testbed.

Classifier	Accuracy (%)
J48	63.97
Naïve Bayes	55.77
NB Tree	66.16
Random Forest	63.26
Random Tree	58.51
Multilayer Perceptron	57.34
SVM	42.29

Table 5. Various Classifiers on NSL-KDD [14]

Second, predictions should become better if we use training information. If we are able to collect our own data, one method of training would be to get information on types of attacks, frequency of attacks, etc. Some of this information was used by other researchers described above. For this improvement, students would need to analyze past data and make predictions. As the predictions are tested, the students would culminate a great learning experience.

Third, we made assumptions when we parsed the data into BoWIDS - no stop words were used, and we removed the top 10% most frequent terms. As we alter these assumptions it is possible that improvements will be found, so student research could permute these possibilities. For example, concerning stop words, none could be used, a few used, and a larger number used. Concerning the frequency of terms, we could consider the top and bottom 0%, 5%, and 10%. Students could form permutations of these choices, run the tests, and see which results are best.

Finally, cluster analysis is one of the most interesting questions. How do we determine which clusters should be labeled INSPECT and which should be NORMAL? If we follow an estimate that on a typical network most traffic is normal (95%) should we choose clusters so that the largest fall in the NORMAL category? In this case, the largest clusters would account for at least 95% of the data. As a future project we will consider other methods for labeling NORMAL and INPECT clusters. It is also possible that cluster selection needs to be incorporated into the clustering algorithm instead of being performed once clustering is complete. This would be research for the most advanced students, perhaps on the graduate level. The student would need to be able to take apart the NMF process and determine how to alter the clustering.

6. Conclusion

Based on a student research project, we have created a prototype called BoWIDS, an intrusion detection system using a Bag of Words model. Various student teams, comprised mostly of undergraduates, studied multiple topics outside of their regular coursework, and wrote code which was used in this prototype. These teams were managed to provide a simulated at-work experience, and hence also improve the students' workplace "*soft skills*" such as team work, oral and written communication, documentation, etc. We base this conclusion on assessment data collected for ABET accreditation.

The strength of BoWIDS is that it can use almost any delimited data. The IDS data is represented by vectors, and the vectors are clustered using Sparse Nonnegative Matrix Factorization. If BoWIDS determines that a vector may represent an attack, the vector is labelled INSPECT, and other vectors are labelled NORMAL. We also compared BoWIDS using NMF with BoWIDS using *k*-means clustering. On the examined data, BoWIDS gave better clusters using NMF because of the size of the clusters, and BoWIDS gave accuracy levels equivalent to methods using training data.

Future improvements to the system will include gathering more data for IDS testing, using training for better accuracy, testing different assumptions in data preparation, and redesigning the cluster analysis. We plan to continue to use student groups to

perform these advancements and thus provide experiential learning opportunities which will heighten students' knowledge and ability in computer science, mathematics, and cybersecurity, as well as improve their soft skills. Overall, we found this to be a great partnership between faculty and student research.

References

[1] Devito, M., Hofmeister, C., Jochen, M.,Schembari. N. P. (2011). Undergraduate research in computer forensics. In Proceedings of the Information Security Curriculum Development Conference (InfoSecCD '11). ACM, New York, NY, USA, 61-68. DOI=10.1145/2047456.2047466 http://doi.acm.org/10.1145/2047456.2047466

[2] Manning, C. D., Raghavan, P.,Schütze, H. (2008). Introduction to information retrieval. Cambridge University Press, New York, NY, USA. Retrieved on 22 July 2015 from http://nlp.stanford.edu/IR-book.

[3] Marszalek., M., Schmid, C., Harzallah, H., van de Weijer. J. (2007). Learning object representations for visual object class recognition. In Proceedings Visual Recognition Challenge Workshop. Retrieved on 16 April from https://lear.inrialpes.fr/pubs/2007/MSHV07/MarszalekSchmid-VOC07-LearningRepresentations-slides.pdf.

[4] Uijlings, J. R. R., . Smeulders,A. W. M., Scha.R. J. H. (2009). Real-time bag of words, approximately. In Proceedings of the ACM International Conference on Image and Video Retrieval (CIVR '09), Article 6, 8 p. DOI = 10.1145/1646396.1646405.

[5] Berry, M. W., Browne, M. (2005). Email surveillance using nonnegative matrix factorization. *Comput. Math. Organ. Theory* 11 (3) (October 2005), 249-264. DOI = 10.1007/s10588-005-5380-5.

[6] MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. *In*: Proc. 5th Berkeley Symp. Math. Statist. Probab., June, p. 281–297. Retrieved on 22 July 2015 from http://projecteuclid.org/download/pdf_1/euclid.bsmsp/1200512992.

[7] Redmond , S. J.,Heneghan,C. (2007). A method for initialising the k-means clustering algorithm using kd-trees. *Pattern Recogn. Lett*. 28, 8 (June 2007), 965-973. DOI=10.1016/j.patrec.2007.01.001.

[8] Lee, D. D., Seung. H. S. (1999). Learning the parts of objects by non-negative matrix factorization. Nature, 401(6755):788-791,. Retrieved on 22 July 2015 from http://www.columbia.edu/~jwp2128/Teaching/W4721/papers/nmf_nature.pdf.

[9] Lee, D. D., Seung,H. S. (2000). Algorithms for non-negative matrix factorization. In Advances in Neural Information Processing 13, MIT Press, 2001. Retrieved on 22 July 2015 from http://www.ccs.neu.edu/home/yzsun/classes/2014Spring_CS7280/Papers/Clustering/NNF_lee01algorithms.pdf.

[10] Pascual-Montano, A., Carazo, J. M., Kochi, K., Lehmann, D., Pascual-Marqui, R. D. (2006). Nonsmooth nonnegative matrix factorization (nsNMF). *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28, 3, 403-415. March 2006. DOI = 10.1109/TPAMI.2006.60.

[11] Kim, J., Park,H. (2008). Sparse nonnegative matrix factorization for clustering. Tech. Rep. GT-CSE-08-01, Georgia Inst. of Technology. Retrieved on 22 July 2015 from http://hdl.handle.net/1853/20058.

[12] KDD Cup 1999 Data. Retrieved on 22 July 2015 from http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[13] Chetan, R., R Ashoka, D.V. (2012). Data mining based network intrusion detection system: A database centric approach. *In 2012 International Conference on Computer Communication and Informatics* (ICCCI 2012), 1, 6, 10-12. January. DOI = 10.1109/ICCCI.2012.6158816.

[14] Lee, W., Stolfo, S.J. Mok, K.W. (1999). A data mining framework for building intrusion detection models. *In* : Proceedings of the 1999 *IEEE Symposium on Security and Privacy*, 120, 132, 1999. DOI = 10.1109/SECPRI.1999.766909

[15] Tavallaee, M., Bagheri, E., Wei, L., Ghorbani, A.A. (2009). A detailed analysis of the KDD CUP 99 data set. IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009 (CISDA 2009), 1, 6, 8-10. July. DOI = 10.1109/CISDA.2009.5356528.

[16] Therdphapiyanak., Piromsopa, K. (2013). An analysis of suitable parameters for efficiently applying k-means clustering to large TCPdump data set using Hadoop framework. In 10th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON 2013), 1, 6, 15-17. May. DOI = 10.1109/ECTICon.2013.6559650.

[17] Zhang, C., Zhang, G. ., Sun, S. (2009). A mixed unsupervised clustering-based intrusion detection model. In 3rd International Conference on Genetic and Evolutionary Computing (WGEC '09), 426, 428, 14-17. Oct. 2009. DOI = 10.1109/WGEC.2009.72.

[18] Kang, D. K., Fuller, D., Honavar, V. (2005) . Learning classifiers for misuse and anomaly detection using a bag of system calls representation. In Proceedings from the Sixth Annual IEEE Information Assurance Workshop, 2005 (IAW '05), p.118, 125, 15-17. June 2005. DOI = 10.1109/IAW.2005.1495942.

[19] Wang, W. , Guan, X., Zhang, X. (2004). Profiling program and user behaviors for anomaly intrusion detection based on non-negative matrix factorization. In 43rd IEEE Conference on Decision and Control (CDC2004), p. 1, 99, 104, 14-17. December. DOI = 10.1109/CDC.2004.1428613

[20] Petrovic, S., Alvarez, G., Orfila, A., Carbo, J. (2006). Labelling clusters in an intrusion detection system using a combination of clustering evaluation techniques. *In*: Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS '06), 6, 129b, 04-07. Jan. 2006. DOI = 10.1109/HICSS.2006.247.

[21] Denatious, D. K., John, A. (2012). Survey on data mining techniques to enhance intrusion detection. In 2012 International Conference on Computer Communication and Informatics (ICCCI 2012), p.1, 5, 10-12. January 2012. DOI = 10.1109/ICCCI.2012.6158822.

[22] The NSL-KDD Data Set. Retrieved on 16 Apr 2015 from http://nsl.cs.unb.ca/NSL-KDD.