Intrusion Detection Based on Feature Transform Using Neural Network

Wonil Kim[1], Se-Chang Oh[2], and Kyoungro Yoon[3]*

[1] Sejong University, Dept. of Digital Contents, 98 Gunja-Dong, Gwangjin-Gu, Seoul, Korea 143-747
wikim@sejong.ac.kr

[2] Sejong Cyber University, 111-1 Gunja-Dong, Gwangjin-Gu, Seoul, Korea 143-747
scoh713@naver.com

[3] Konkuk University, Dept. Computer Science and Engineering, 1 Hwayang-Dong, Gwangjin-Gu, Seoul, Korea 143-701
yoonk@konkuk.ac.kr

Abstract. In this paper, a novel method for intrusion detection is presented. The presented method uses a clustering method based on the transformed features, which can enhance the effectiveness of the clustering. Clustering is used in anomaly detection systems to separate attack and normal samples. In general, separating attack and normal samples in the original input space is not an easy task. For better separation of the samples, a transformation that maps input data into a different feature space should be performed. In this paper, we propose a novel method for obtaining proper transformation function that reflects the characteristics of the given domain. The transformation function is obtained from the hidden layer of the trained three-layer neural network. Experiments over network connection records from KDD CUP 1999 data set are used to evaluate the proposed method. The result of the experiment clearly shows the outstanding performance of the proposed method.

1 Introduction

The goal of an Intrusion Detection System (IDS) is to detect any kind of attack effectively, whereas reducing false alarm of legitimate accesses. There have been many researches on this field and published various methods to achieve this goal, but not many of the previous systems reported satisfactory results on achieving the goal of separating attacks and legitimate accesses. There are two categories for intrusion detection method. One is misuse detection method and the other is anomaly detection method. In the misuse detection method, a model called a signature is developed based on the attack patterns and any activity similar to the defined signature is regarded as an intrusion [1]. On the other hand, in the anomaly detection method, a model called a profile is developed based on the normal access patterns, and any activity that deviates from the defined profile is regarded as intrusion [2].

* Author for Correspondence, +82-2-450-4129. This work was supported by the faculty research fund of Konkuk University in 2003.

M. Bubak et al. (Eds.): ICCS 2004, LNCS 3037, pp. 212–219, 2004.

There are several ways to model the legitimate behavior, such as statistical method, data mining, and clustering. In statistical method, statistical values like mean and variance are calculated for features of data, and the set of these values is used as a profile. In [3][4], a statistical distribution is assumed and the parameters for this distribution is estimated from data. Statistical method is too simple to model complex distributions of data. Another approach is using data mining technique. It finds common patterns in given data, after which a decision tree or a rule set is constructed from the discovered common patterns [5]. However, the complexity of the method is too high to be used in a real-time application. Finally, clustering selects several representative centroids for normal samples in the input space [6]. Then the selected centroids are used as the profiles. The clustering can be performed in unsupervised manner since most of the data is normal [7]. This method can model any data of complex distribution, and the computational cost is acceptable. The problem of reported clustering approaches [6][7] is that the distribution of normal and attack samples is not easily separable and in some cases, the normal data themselves are not well clustered.

This paper presents a novel approach based on the clustering methods to solve the problems, in which the normal and attack samples are not separable and the samples themselves are not well clustered. We solve the clustering problem by using transformed features instead of the original input fields (attributes). The original data are transformed to a new feature space so that the attack and normal samples are well clustered and clearly separable. The feature transform is performed using the hidden layer nodes of trained neural network as explained in section 3.2 and 4.3..

The main problem and main idea are described in section 2 and section 3 respectively. The experimental results using KDD cup data are shown and evaluated in section 4. Finally, the conclusion and discussion are given in section 5.

2 Problem Description

The problem is to find a reasonable transformation function that separates samples from different classes and categorizes the samples into the same class.

In the proposed approach, the feature space of the original given data is transformed into a different feature space. There are several reasons for this transformation. First, there may be highly correlated data fields in the raw data [8]. Principal Component Analysis (PCA) [9] and Independent Component Analysis (ICA) [10] that find independent features in the given data are some of the most popular methods to find such data fields in the raw data. Second, several data fields dominate the rest of the data fields as the raw data is usually not normalized. Generally, balancing the fields by weighting is done manually in preprocessing. Third, the most radical reason is that samples from different classes, in general, are not separable in the input space. This appears commonly in many problem domains, and makes the clustering difficult and inefficient. The first goal of the presented approach is finding a transformed feature space in which the samples are well separable and grouped.

3 Data Transformation and Anomaly Detection

There are two steps in this data transformation. The first step is finding fields irrelevant in discriminating two classes of data, i.e. attack and normal classes, and excluding the irrelevant fields. Of course, a proper transformation function would ignore these fields even if they are not excluded beforehand. But if we can exclude them beforehand, it would reduce the computational costs. The second step is finding a transformation function itself. In this paper, we propose a method for finding a transformation function using neural networks. It exploits the fact that a trained neural network has capability of discriminating between classes. We discuss about how to use a neural network trained with a given set of labeled samples for data transformation in the following sections. Through the two steps, input data are expected to be transformed into a well separable feature space.

After the input data are transformed into the desired feature space, we use *k*-means algorithm [11] to define clusters of normal class data including the centroid and radius of each cluster. If new data (data under test) are not within the radius of these clusters, they will be considered as attack data.

3.1 Field Selection

In this process, the fields whose values are scarcely or randomly changed are removed from the feature space using three steps, as they are either meaningless in the aspect of information theory, or not helpful for discriminating between classes.

In the first step, the fields whose values hardly change are selected to be removed, as the fields with relatively fixed value do not give any discriminating information. In the second step, the fields whose values change randomly are selected to be removed, as the random behavior makes it very hard to find rules for discriminating classes. To measure the randomness, entropy of each field is calculated. Fields with high entropy values are considered as random fields. In the third step, Bayes decision theory is used to evaluate the contribution of each field to the discrimination between classes. For this step, the Bayesian threshold value for each field, is selected based on the normal and attack samples from the labeled training data set from KDD Cup 1999 [12]. Using the selected threshold value, the error probability of using each field for the discrimination factor is calculated. Finally, the fields with high probability of error are removed.

3.2 Input Data Transformation

In this step, we employ neural network to obtain the function $\boldsymbol{\tau}$ for transforming data. The well-known back propagation algorithm is used to train the neural network [13]. Using the weights in the trained neural network, we can obtain the transform function. The transformation is a process for obtaining the feature vector $\boldsymbol{b}$. As a result, we obtain the transformation function $\boldsymbol{b} = \boldsymbol{\tau}(\boldsymbol{a})$ where $\boldsymbol{a}$ is the input vector (selected

features from the original input space) and $\boldsymbol{b}$ is the output vector acquired from input to the output node of the trained neural network.

In other words, a neural network with hidden nodes is trained, and the values of the hidden nodes in the trained neural network provide the parameters for the feature transformation.

As the training of the neural network is performed with approximately the same number of normal samples and attack samples, less than 2 percent difference to be exact, the trained neural network models both normal pattern and attack pattern. Therefore, the proposed method is not biased to either normal samples or attack samples in the process of developing the transform function, and provides effective discrimination power of normal and attack samples.

3.3 Clustering and Class Labeling

Once the transformation function is acquired, i.e. the neural network is trained, profile of the normal data is ready to be developed using clustering method.

There are various clustering methods available, and we use the *k*-means algorithm for our clustering method [11]. Assuming that most of the network connections are normal patterns in practice, we can build the clusters only with normal dataset. When *k* clusters are built, we remove clusters with relatively small number of samples. In the process of cluster removal, the assumption is that the normal samples in the transformed feature space are well grouped together and most of the clusters are practically empty when *k* is large enough. The appropriate *k* (number of initial clusters) and the number of remaining clusters after removal can be decided experimentally. The definitions of the remained clusters become the profile for the anomaly detection.

When a connection record is received from the network, we can acquire transformed features of the connection record using the trained neural network, i.e. using the developed transform function. The transformed feature vector is compared to the cluster definitions in the profile. If it does not belong to any of the clusters in the profile, the connection is considered anomalous. In other words, any samples which do not belong to any of the developed clusters of the normal samples are considered attack samples. This idea of using limited number of clusters in the transformed feature space outperforms any known anomaly detection system as shown in the experimental results.

4 Experimental Results

4.1 Data Description

We evaluated our proposed method over the KDD Cup 1999 data [12]. It contains a wide variety of intrusions simulated in a military network environment. Each sample in the data is a record of extracted features from a network connection gathered during the simulated intrusions. A connection is a sequence of TCP packets to and from various IP addresses. The TCP packets were assembled into connection records using

the Bro program [14]. A connection record consists of 42 fields. It contains basic features about TCP connection as duration, protocol type, and number of bytes transferred. It contains domain specific features as number of file creation, number of failed login attempts, and whether root shell was obtained. Also it has the last field for labeling it as either normal or one specific kind of attack. Following is a typical example of connection record.
(0,udp,private,SF,105,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,
0.00,0.00,255,254,1.00,0.01,0.00,0.00,0.00,0.00,0.00,0.00,normal)

We used three data files for experiments: the file named "kdd-train-nor" and "kdd-train-att" for training the neural network and for developing normal clusters; and the file named "kdd-test" for evaluation. The origins of these files are "kddcup.data_10_percent" and "corrected" downloaded from KDD Cup 1999 site. The file "kdd-train-nor" consists of 97,278 normal samples selected from the file "kddcup.data_10_percent". The file "kdd-train-att" consists of 95,649 attack samples selected from the file "kddcup.data_10_percent" so as to make a balance in size with the training set of normal samples. Finally, the file "kdd-test" which is the same with the file "corrected" consists of labeled 60,593 normal and 250,436 attack samples.

4.2 Field Selection

Among the 42 fields of connection record, the symbolic fields such as protocol_type, service, and flag are difficult to represent in numerals. A Bayesian classifier is trained and tested for each of 38 fields except three symbolic fields and the label field. Fig. 1 shows the error rate of Bayesian classifier for each field. Any field with the error rate higher than 20%, as a rule of thumb, is removed and 12 fields are selected.

4.3 Data Transformation

A three-layered neural network was used to acquire transformation function for the simplicity of computation. The input layer of the neural network consists of 12 nodes to present 12 field values as input. The number of nodes in the middle layer (hidden layer) should be decided depending on the complexity of the problem. As the number of node increases, the complexity of the network increases. Also large number of nodes makes the train time longer and the resulting neural network could be over-specialized. If the number is too small on the other hand, the neural network cannot be trained as we wanted. In this intrusion detection system, ten nodes are employed for the middle layer, based on the experiments. Finally, the output layer has only one node because the number of classes to be distinguished is two (normal or attack). In this case, if the value of output node is close to 1, the given data is accepted to belong to the normal class and if the value is close to 0, the given data is accepted to belong to the attack class. The neural network is trained using the data set "kdd-train-nor" and "kdd-train-att" as explained in section 4.1.

After the training of the neural network, each sample is transformed before clustering is performed. The equation $\boldsymbol{b} = \tau(\boldsymbol{a})$ is computed using the weights of the

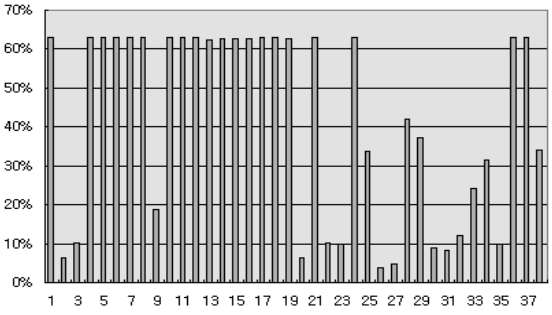

Fig. 1. The error rate of Bayesian classifier for each field is shown. Among 38 fields, Bayesian classifiers of only 12 fields have error rate less than 20% and those 12 fields are selected as input space features.

first two layers from the trained neural network to transform a sample. In other words, the two layer neural network with 10 output nodes (the hidden nodes in the original three layer neural network), which is acquired by removing the single node output layer from the trained neural network, represents the transformation function.

4.4 Clustering and Anomaly Detection

We used the file "kdd-train-nor" as the training set to build up the cluster set, since we assumed that most of the samples are normal in practice, following the anomaly detection approach. As a result, each sample $\boldsymbol{a}$ of 12-dimensional vector from the file "kdd- train-nor" is transformed to a 10-dimensional feature vector $\boldsymbol{b}$. Then the *k*-means algorithm [11] is used for clustering the set of transformed vectors. It is possible to measure the size of each cluster by counting the number of samples classified as the cluster. The clusters with small size are removed from the cluster set, after which the remaining clusters contain 98% of the original samples. The centroids and radii of the remaining clusters are used as the profile of the normal data.

Fig. 2 shows the number of remaining clusters after removing small clusters. The horizontal axis stands for the value *k* which is the number of initial clusters. The result of clustering with original data shows that, as the number of clusters increase, more normal samples are covered by the clusters. In other words, the samples are scattered in the original input feature space. On the other hand, the result of clustering with transformed data shows that the number of clusters stabilizes rapidly. This means that the samples are well grouped in the transformed feature space.

Fig. 3 (a) and Fig. 3 (b) show the classification result with and without transformation. Two indicators are used to measure the performance: the attack detection rate and the false positive rate. The file "kdd-test" that contains both normal samples and attack samples are used as the test set. Cluster information (profile of normal data) acquired using the *k*-means algorithm with "kdd-train-nor" data set in the previous stage is used for classification task. Fig. 3 (a) shows that it is hard to increase the

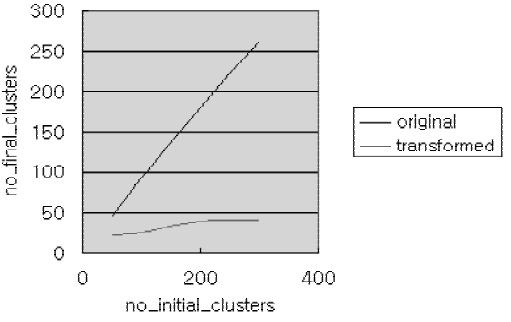

Fig. 2. Even if the number of initial clusters (k) increases, most of the samples from the normal connection stay within very limited number of clusters in the transformed feature space. In this experiment, less than 50 of the clusters cover most of the normal samples from the KDD Cup dataset.

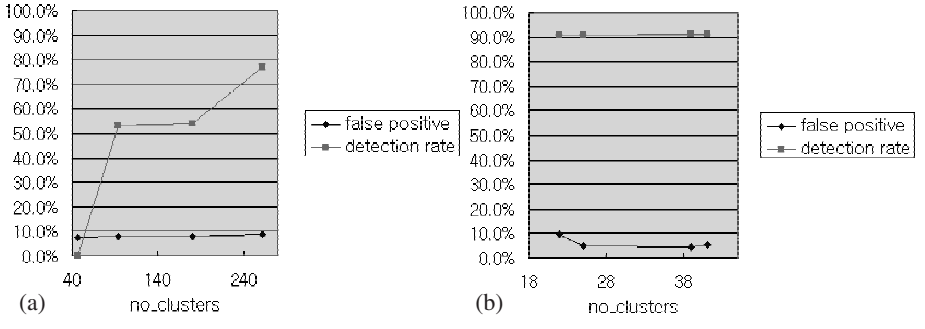

Fig. 3. Theses figures compare the performance of anomaly detection with various numbers of clusters. (a) and (b) show the performances when the clustering is performed in the input-feature space and the transformed feature space, respectively. In (a), we can clearly see that the detection rate depends on the number of clusters used. In other words, the samples are scattered in the original input feature space and they cannot be covered by a small number of clusters. On the other hand, in (b), we can clearly see that less than 40 clusters are enough to cover most of the normal samples and the normal samples are well clustered together. Furthermore, when the feature space transformation is performed as presented, the detection rate greatly increases, while the false positive rate is kept low using only a small number of clusters.

detection rate, even if we increase the number of clusters, when the transformation is not performed. On the other hand, Fig. 3 (b) clearly shows that the detection rate stabilizes to a higher level with relatively small number of clusters, whereas the false positive rate is kept very low, when the transformed data are used.

5 Conclusion

In this paper, we proposed a neural network based feature transformation for intrusion detection using clustering. First, we explained how the characteristics of data can affect the clustering results. Then we discussed how the raw data should be trans-

formed in order to achieve efficient clustering. To evaluate the proposed method, the performance of anomaly detection based on the clustering in the transformed feature space was compared with the performance of anomaly detection based on the clustering in the input space. The result shows that the number of generated clusters is decreased and the correctness is increased dramatically in the transformed feature space. In practice, the number of clusters critically affects the efficiency of intrusion detection engines. The proposed method, in this sense, clearly improves the efficiency as well as the correctness of the intrusion detection system. This method can be easily extended to many problems in which the samples in the original feature space is not easily separable.

References

1. Lunt, A., Tamaru, A., Gilham, F.: IDES: A Progress Report. Proceedings of the 6th Annual Computer Security Applications Conference Tucson, AZ (1990)
2. Denning, D.: An Intrusion Detection Model. Proceedings of the Seventh IEEE Symposium on Security and Privacy (1986) 119-131
3. Ye, Nong: A Markov chain model of temporal behavior for anomaly detection. Proceedings of the 2000 IEEE Systems, Man, and Cybernetics; Information Assurance and Security Workshop (2000)
4. Eskin, E., Lee, W., Stolfo, S. J.: Modeling system calls for intrusion detection with dynamic window sizes. Proceedings of DARPA Information Survivability Conference and Exposition II (DISCEX II) (2001)
5. Lee, W., Stolfo, S. J., Mok, K.: A Data Mining Framework for Building Intrusion Detection Models. Proceedings of 1999 IEEE Symposium on Security and Privacy (1999)
6. Portnoy, L., Eskin, E., Stolfo, S.: Intrusion detection with unlabeled data using clustering. Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA 2001) (2001)
7. Eskin, E., Arnold, A., Prerau, M., Portnoy, L., Stolfo, S.: A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data. Data Mining for Security Applications, Kluwer (2002)
8. Dura, R. O., Hart, P. E.: Pattern Classification and Scene Analysis. Wiley-Interscience Publication (1973) 247
9. Tou, J., Heydorn, R.: Some Approaches to Optimum Feature Extraction. Computer and Information Sciences II, Academic Press, New York (1967)
10. Lee, Te-Won: Independent Component Analysis: Theory and Applications. Kluwer Academic Publishers (1998)
11. MacQueen, J.: Some Methods for Classification and Analysis of Multivariate Data. Proceedings of the 5th Berkeley Symposium on Probability and Statistics, University of California Press, Berkeley (1967)
12. The third international knowledge discovery and data mining tools competition dataset KDD99-Cup. Available at `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html` (1999)
13. K.Mehrotra, C.K.Mohan, and S.Ranka, Elements of Artificial Neural Networks, MIT Press (1997)
14. Paxon, V.: Bro: A system for detecting network intruders in real-time. Proceedings of the 7th USENIX Security Symposium, San Antonio, TX (1998)