Generating Artificial Attack Data for Intrusion Detection Using Machine Learning

Truong Son Pham
Faculty of IT
Le Quy Don University
Hanoi, Vietnam
sonpham.mta@gmail.com

Quang Uy Nguyen
Faculty of IT
Le Quy Don University
Hanoi, Vietnam
quanguyhn@gmail.com

Xuan Hoai Nguyen
IT Research Centre
Hanoi University
Hanoi, Vietnam
nxhoai@gmail.com

ABSTRACT

Intrusion detection based upon machine learning is currently attracting considerable interests from the research community. One of the appealing properties of machine learning based intrusion detection systems is their ability to detect new and unknown attacks. In order to apply machine learning to intrusion detection, a large number of both attack and normal data samples need to be collected. While, it is often easier to sample benign data based on the normal behaviors of networks, intrusive data is much more scarce, therefore more difficult to collect. In this paper, we propose a novel solution to this problem by generating artificial attack data for intrusion detection based on machine learning techniques. Various machine learning techniques are used to evaluate the effectiveness of the generated data and the results show that the data set of synthetic attack data combining with normal one can help machine learning methods to achieve good performance on intrusion detection problem.

Keywords

Intrusion Detection, Artificial Attack, Machine Learning

1. INTRODUCTION

Network-based intrusion detection system (NIDS) has played an important role in network security due to the widespread use of computer network, the increase in valuable resources and the rapid development of attackers [17]. Nevertheless, traditional signature-based intrusion detection techniques have failed to fully protect networks and systems from increasingly sophisticated attacks and malwares. Consequently, machine learning based intrusion detection systems have become an indispensable component of security infrastructure used to detect these threats before they inflict widespread damages [26].

When building a machine learning based NIDS one needs to consider many issues, such as data collection, data preprocessing, intrusion recognition, reporting, and response [29]. Among them, data collection and intrusion recognition are perhaps at the heart. Since Denning first proposed an intrusion detection model in 1987 [10], the research efforts have been focused on how to accurately construct detection models. While most previous research paid attention to enhance the accuracy of learning methods that were evaluated on some available data set such as KDD Cup 1999 [26], the research on how to effectively collect the training data for a real NIDS seems much less [26].

In 1999, KDD Cup 99 (referred as KDD'99 hereafter) data set was released and used for The 1999 Knowledge Discovery and Data Mining Tools Competition [18]. This data set was constructed from DARPA data set of MIT Lincoln Laboratory in an attempt to evaluate intrusion detection techniques for their systems [1]. Over the last decade, KDD'99 has been extensively used by many researchers as the benchmark to evaluate their learning algorithms. The main advantage of KDD'99 data set is that it provides an easy access and a common benchmark for NIDS evaluation. However, the downside of this data set is that it has been out of date and not suitable for building a NIDS in reality [5, 25].

In order to construct a detection model for a real IDS, it is often required that the training data is up to date and as close as possible to the data in the real environment. In other words, such data sets (both intrusive and normal data) usually need to be acquired from the real interested networks. While the benign data can be relatively easy to sample from the normal behaviors of the networks, collecting intrusive data is often more difficult and expensive. In this paper, we propose a solution to this problem by introducing some schemes to generate artificial attack data and evaluate the effectiveness of this data by various machine learning techniques. The experimental results show that most learning methods can achieve good performance on the artificially generated data.

The remainder of the paper is organised as follows. In the next section, we give a short introduction to machine learning with the emphasis on the methods used in this paper. After that, we briefly review some previous research on intrusion detection using machine learning techniques. In Section 4 we present our methods to generate artificial/synthetic attack data for training learning machines to solve the intrusion detection problem. It is followed by a section detailing our experimental settings. The experimental results are shown and discussed in Section 6. The last section concludes the paper and highlights some potential future work.

SoICT '14 December 04 - 05 2014, Hanoi, Viet Nam

http://dx.doi.org/10.1145/2676585.2676618

[1]This data set is available at: http://www.ll.mit.edu/mission/communications/ist/index.html

2. BACKGROUNDS

Machine learning concerns the construction and study of systems that can learn from data [4]. This can be seen as one of the main branches of artificial intelligence. There are two important issues in machine learning: representation and generalization. Representation considers the way in which data instances are described and the format of the learnt models. Generalization is the property that the system will perform well on unseen data instances. This perhaps is the most important objective of all machine learning systems [21].

Machine learning algorithms can be categorized into three major subclasses based on the type of input available during training of the machine [4]. The first subclass is supervised learning algorithms where all training samples (the inputs) are labeled. The supervised learning algorithm attempts to construct a function or a mapping from inputs to outputs which can then be used to generate an output for previously unseen inputs [16]. The second subclass is unsupervised learning algorithms that operate on unlabeled examples. In other words, the desired output of these sample is unknown. Here the objective is not to find a mapping to inputs to ouput but to discover the salient structure in the data through some techniques like cluster analysis [3]. Finally, the third subclass is semi-supervised learning algorithms that aim to use both labeled and unlabeled examples to gain better performance [16].

Among three subfields of machine learning methods, supervised learning algorithms are widely recognized as the most popular form. Various supervised learning algorithms have been proposed with numerous applications in different areas. In this paper, we will use some well-known supervised learning techniques to evaluate the effectiveness of the method to generate the synthetic attack data. These algorithms are briefly described below.

Support Vector Machines: Support Vector Machines (SVM) is a relatively new learning method used for binary classification [8]. Since its introduction by Vapnik in 1998 [27], SVM has successfully been applied to many real-world problems [8]. SVM first maps the input vector into a higher dimensional feature space using a kernal function. After that, an algorithm is used to obtain the optimal separating hyperplane in the higher dimensional feature space. Moreover, a decision boundary, i.e. the separating hyper-plane, is determined by support vectors, the training samples that are close to a decision boundary, rather than the whole training samples and thus SVM is extremely robust to outliers. The SVM also provides a user specified parameter called penalty factor that are adjusted to make a trade off between the number of misclassified samples and the width of the decision boundary.

Artificial Neural Networks: An Artificial Neural Network (ANN) is an information processing paradigm aiming to mimic the way that the biological nervous system processes information [9]. An ANN consists of a collection of processing units called neurons that are highly interconnected according to a given topology. Similar to human beings, ANNs have the ability of learning-by-example and generalization from limited, noisy, and incomplete data. ANNs have been successfully employed in a broad spectrum of real world applications such as data classification and pattern recognition. There are several different kinds of neural networks, of these, RBFNetwork (Radial Basis Function Network) and Multilayer Perceptron [28] are perhaps the most popular. These networks will be used in the experiments in this paper.

Bayesian networks: A Bayesian network is a model that encodes probabilistic relationships among the variables of interest [15]. A Bayesian network (BN) consist of nodes and arcs that present for random variables and connections between them, respectively. When constructing a network, two main components namely estimator and searching algorithm need to be identified. While estimator is used to evaluate the performance of a given network, searching algorithm aims to search through the space of possible networks to find a good candidate network. In our experiments, we used SimpleEstimator algorithm for the estimator, and K2 for the searching algorithm. They were both implemented in Weka [28].

Naive Bayes Network: A naive Bayes network classifier is a simple probabilistic classifier based on applying Bayes' theorem with strong (naive) independence assumptions [30]. In reality, there are some cases where the presence or absence of a particular feature is unrelated to the presence or absence of any other feature, given the class variable. In order to exploit this structural relationship or casual dependencies between the random variables of a problem, naive Bayesian networks can be used. Despite oversimplified assumptions, naive Bayes network classifiers have worked quite well in many complex real-world situations. In this paper, we also use naive Bayes network to classify between intrusive and normal data.

Decision trees: Decision tree learners are a well-established family of learning algorithms for classification [23]. A decision tree classifies a sample through a sequence of decisions, in which the current decision helps to make the subsequent decision. In contrast to other back-box learning methods, the sequence of decisions is represented in a white-box structure (tree structure). This allows its solutions can easily be understood and analysed. The classification of a sample is started from the root node to a suitable leaf node, where each leaf node represents a classification category. There have been a number of algorithms developed for constructing decision trees for a problem. Among them, C4.5 and its variants are the most popular algorithms [24]. In this paper, we use an extension of C4.5 called J48 for constructing the decision tree for intrusion detection problem.

3. RELATED WORK

Machine learning for intrusion detection has received increasingly attention in the research community [29]. Diverse learning techniques have been proposed to tackle the intrusion detection problem. These methods can be generally divided into three categories, namely single, hybrid, and ensemble methods [29]. The single method attempts to use only one machine learning technique such as artificial neuron networks or support vector machines to find the models which are used to classify or recognize whether the incoming Internet access is the normal access or an attack [6]. In the single method, K-NN and SVM are the most commonly used techniques for intrusion detection [19] with SVN often produces good performance [6].

The hybrid method aims to combine some learning techniques to enhance the performance of the systems. There are several ways in which different algorithms can be hybridized. The hybrid classifiers can be built based on cascading different classifiers, such as neuron-fuzzy techniques where the

outputs of the first classifiers are severed as the inputs for the second classifiers [7], and so on. On the other hand, hybrid classifiers can be implemented by using some clustering-based approaches to preprocess the input samples in order to eliminate unrepresentative training examples from each class. Then, the clustering results are used as training examples for classifier design [20]. Finally, hybrid classifiers can also be constructed on the integration of two different techniques in which the first aims at optimizing the learning performance (i.e. parameter tuning) of the second model for prediction [1].

The third method uses ensemble learning techniques to improve the classification performance of a single classifier [13]. Ensemble learning aims to combine multiple weak learning algorithms or weak learners to create the problem solution. There are several strategies for combining weak learners, among them, the "majority vote" is arguably the most commonly used method in the literature. Other combination methods, such as boosting and bagging, are also popular. These methods are based on resampling data samples and then taking a majority vote of the resulting weak learners [22].

Although, numerous machine learning algorithms have been used for intrusion detection, most of these research considered some public data sets like KDD'99 or DARPA 1998 for their experiments [26]. These public data sets are recognized as standard data sets in intrusion detection though they might not be suitable to find detection models for an IDS in reality. There were also some studies using non-public or their own data sets [25]. However, the number of these researchers is much smaller compared to those using KDD'99 or DARPA 1998. The reason could be that collecting the training data, especially intrusive data, is often difficult and expensive. This paper aims to address this problem by proposing methods for creating artificial attack data. These methods are detailed in the following section.

4. METHODS

This section presents our main methods to generate synthetic intrusive data for machine learning approaches. There are two situations where this artificial attack data may be beneficial for learning systems. The first situation is when no intrusive examples can be collected. In other words, only benign samples are gathered from the normal behaviors of the networks. In this case, the attack data will be generated based on the previously collected normal data. In the second situation, we can collect some intrusive examples but the number of these samples are rather small. That is to say collected intrusive data is not enough to build a good detection model. In this case, the artificial intrusive samples will be created with respect to the previous obtained attack examples. Two methods for generating intrusive data in two these situations are described below.

The idea behind the method for generating intrusive data in the situation where none of the attack data has been collected is that attack behaviors are often by far different from normal behaviors. Therefore, if we create some samples that are very much far from the samples in the normal data set, then these samples can be seen as the abnormal (intrusive) data. The method for generating this kind of data is presented in Algorithm 1.

The input of the algorithms is the normal data set, D, with N samples, and the number of features is F. It is also important to note that all feature values of D must be converted into numerical format before the algorithm can be executed. D' is the intrusive data set generated with N' samples of F features. The algorithm begins by calculating mean and standard deviation of each feature value in D. After that, a number of samples (N') in D' are generated by randomly selecting a sample in D, copying it to D' and altering every value of its features by a new value so as the generated sample is different enough from the samples in D (values of the features of the new samples is out of the range $[\mu - 3\delta, \mu + 3\delta]$). This is based the assumption that all features of normal data follow the normal distribution. If so, a value that is out of the range $[\mu - 3\delta, \mu + 3\delta]$ will often be seen as generating by an abnormal behavior [2].

Algorithm 1: Generating Synthetic Attack Data from Normal Data

```
Input is the normal dataset: D
Output is the intrusive data set: D'
Calculate mean (μ) and standard deviation (δ) of each
feature value in D
D'=empty
Count=0;
N' is the number of required samples in D'
while Count<N' do
    Randomly select a sample in D called this as S
    Copy S to create a sample S'
    for each feature t in S' with μ_t and δ_t do
        Randomly generate a real value r so that
        r ∉ [μ_t − 3δ_t, μ_t + 3δ_t]
        Replace the value of feature t with r
    Add S' to D'
    Count=Count+1
```

The method for generating intrusive data in the second situation is based on few attack samples that have been obtained previously. The intuition is that the examples of attack data in the future are often similar to those samples in the past though they may not be identical. The algorithm for creating intrusive data in this situation is presented in Algorithm 2. Here the input data set is the intrusive data instead of normal data as in Algorithm 1. Moreover, a new sample is generated by copying and modifying a feature but the margin of modification is kept small enough (the value belongs to the range $[min, max]$ where min and max are the smallest and greatest values among all samples of the selected feature in D'). This is to guarantee that the sample is similar to the previously collected samples in the intrusive data set.

More precisely, in Algorithm 2, a new synthesis sample data is generated as follows. A feature (F) and a sample (S) in D' are randomly selected. Then, the highest frequency of values in F (V_{max}) and the frequency of the value of feature F in S are (V) calculated. After that, V_{max}-V new artificial sample are generated by copying S to D' and altering the value (t) of feature F at S so that this value belongs to $[min, max]$. This algorithm is similar to and inspired from the algorithm to generate artificial data for Anomaly detection in [11]. The difference is the range of the generated random values. In this paper, these values must be in the range of $[min, max]$ rather than in the arbitrary range in [11].

Algorithm 2: Generating Artificial Attack Data based on the Previous Intrusive Data

```
Input is the previous intrusive data set: D'
Output is the artificial intrusive data set: E'
E'=empty
Count=0;
N' is the number of required samples in E'
while Count<N' do
    Randomly select a feature F in D'
    Set min=the smallest value of samples in F
    Set max=the greatest value of samples in F
    Calculate the frequency of each value in F
    Set V_max= the highest frequency of values in F
    Randomly select a sample in D' called this as S
    Set V=the frequency of the value of S in F
    Set t=value of feature F in S
    for i=V to V_max do
        Copy S to create a sample S'
        Randomly generate a real value r so that
        r ⊆ [min, max]
        Replace the value of t with r
        Add S' to E'
        Count=Count+1
        if Count>=N' then
            break;
```

5. EXPERIMENTAL SETTINGS

This section presents the settings for the experiments in this paper. In order to investigate the effective impact of the synthetic data sets, three sets of experiments were set up. The first aims to examine the efficiency of the created data set when none of the attack data was obtained. The second experiment plans to examine if generating more intrusive data from a small amount of attack data can help to improve the prediction ability of learning systems. The third experiment will examine if by generating intrusive data for one kind of attacks can help to detect other attacks better.

In the first set of experiments, 500 normal samples were selected from the KDD'99 data set. After that 500 intrusive data samples were generated using Algorithm 1. These two data sets were combined to become the training data for all learning algorithms. A testing data set of 1500 samples was also chosen from KDD'99 in which 1000 samples were normal data and 500 samples were intrusive data of Distributed Denial-of-service (DDOS) attack [2].

In the second experiment, 400 normal data samples and 200 DDOS attack samples were chosen. After that 200 intrusive samples were generated using Algorithm 2 based on 200 DDOS samples. Two training data set were created. The first includes 600 samples (400 normal and 200 DDOS) and the second has 800 samples (400 normal, 200 DDOS and 200 artificial generated). A testing data set includes 300 DDOS samples and 600 normal samples that were selected from KDD'99.

The third set of experiments aims to address the question whether generating more intrusive data of one attack type can help to detect other attacks better. First, 87 samples of a DDOS attack type–smurf attacks and 400 normal data samples were drawn from KDD'99. After that 87 artificial data was created using Algorithms 2. Similar to the second experiment, two training data set were created. The first includes 487 samples (400 normal and 87 DDOS smurf) and the second includes 574 samples (400 normal, 87 DDOS smurf and 87 artificial samples). The testing includes 1200 samples in which 800 samples are normal data and 400 samples are DDOS data but they are different from the data used for training (smurf). 400 DDOS samples in the testing set comprise of the following attacks: land, back, neptune, pod, teardrop.

The effectiveness of the generated data was examined based on several well-known machine learning techniques. These techniques were trained and tested on the above data sets. The tested learning techniques include decision trees (using J48 algorithm), support vector machines (using SMO algorithm), artificial neural networks (Multilayer Perception and RBF networks) and Bayes Networks (BayesNet and NavieBayes). The results of applying the learning methods to the above data sets will be presented in the following section.

[2]In this paper, we only used DDOS attack data since this is the most popular attack in KDD'99.

6. RESULTS AND DISCUSSION

This section presents the experimental results of applying machine learning techniques to the generated artificial data sets. To apply learning methods to these data sets, we use their implementations in Weka. The parameters of these algorithms are tuned using the method in [28] and the best parameters was selected to report the results. The performance of each algorithm is measured by the percent of correct classification on the testing data set. They are shown in the following tables.

Table 1 presents the results when applying different machine learning methods to the problem of intrusion detection in case none of attack data has been collected. In this table (and the following tables), the first row presents various tested learning techniques including decision trees (J48), support vector machine (SVM), two artificial neural networks (Multilayer Perceptron: Perc and Radial basis function network: RBF), and two Bayesian networks (Bayes Network: Bayes and NavieBayes: Navie). The second row "None" shows the results of learning system when no attack data was used and the last row "Arti" presents the results when artificial intrusive data was generated.

Table 1: The percent of correct classification of learning methods when none of attack data was collected.

Methods	J48	SVM	Perc	RBF	Bayest	Navie
None	66.7	66.7	66.7	66.7	66.7	66.7
Arti	97.1	93.6	96.3	68.2	96.9	97.4

It can be seen from this table that generating synthetic attack data helps all machine learning methods to improve their performance. The table shows that all learning systems perform ineffectively when none of the attack data was obtained. In fact all techniques classify the samples into the one class (the normal class) if there was not any intrusive data was provided to the training process. Therefore the percent of correct prediction of them is identical at 66.7%.

Conversely, the performance of all learning methods are

improved by adding the artificial intrusive data to the training data set. Especially, some methods such as decision trees (J48) and naive Bayes can achieve very good results (the percent of correct prediction is up to nearly 98%). This results show that the method to generate artificial attack data in Algorithm 1 may be suitable for machine learning methods to gain good performance on intrusion detection problem when no intrusive data has been collected.

Table 2 shows the results of machine learning techniques when there are some but few collected attack data. In this table the second row "Few" presents the results when the training set contains few collected intrusive data and the third row "Arfi" shows the results when artificial attack data generated by Algorithm 2 was added to the training data.

Table 2: The percent of correct classification of learning methods when few of attack data was collected.

Methods	J48	SVM	Perc	RBF	Bayest	Navie
Few	90.3	98.2	98.6	93.6	96.6	98.4
Arti	99.0	99.2	99.7	99.2	97.8	99.7

It can be seen from this table that by adding some few collected attack data to the training set, the performance of all learning techniques are improved. These algorithms can achieve up to 98% of correct classification on this problem. The result is mostly equal to the result obtained when using artificial attack data in Table 1. However, what is more important is the performance of all machine learning methods are also enhanced when incorporating the training data with some synthetic intrusive data. Particularly, some methods such as Multilayer Perceptron network and Naive Bayes can classify almost 100% correctly (the percent of correct prediction of both methods are 99.7%). This results show that generating artificial intrusive data using Algorithm 2 is beneficial for the efficiency of the learning techniques.

The last table, Table 3, shows the results of applying the learning techniques to intrusion detection when the objective is to detect new attacks. In this table, the second row "Old" presents the results of machine learning when they are trained on the training set of smurf attack and predicted on other attacks (land, back, neptune, pod, teardrop) and the third row "Arfi" shows the results when the training set is added some artificial data generated by Algorithm 2.

Table 3: The percent of correct classification of learning methods when tested on new attack types.

Methods	J48	SVM	Perc	RBF	Bayest	Navie
Old	67.1	67.1	69.3	67.5	89.4	65.9
Arti	67.6	67.3	67.1	98.6	97.5	97.5

It can be observed from this table that all learning methods are suffered from difficulty in predicting new attacks. Their performances are often not as good as when predict the same types of attacks as in Table 2. Most of the algorithms achieve below 70% correct prediction with the exception on Bayesian network where the correct prediction is 89%. On the other hand, generating synthetic attack data and adding it to the training set often helps machine learning methods detect new attacks better. Particularly, some methods such as RBF network, Bayesian network and navie bayes can achieve very good performance in identifying new type of intrusions: the percent of correct prediction is up to nearly 99%. Overall, the results in this section show that generating artificial attack data gives positive impact on learning systems in both situations when no intrusive data or few intrusive data is collected.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we addressed the problem that practitioners often face when building an intrusion detection system in reality. The problem is the shortage of the training data for machine learning methods. For learning techniques perform effectively on this problem, it is often required that a large amount of data was collected from the real environments. While collecting normal data is often straightforward and inexpensive, obtaining attack data is usually harder and more expensive. This paper proposed some methods for generating attack data to relieve this problem. Particularly, two algorithms were proposed to create intrusive data in two situations when none and few of intrusive data has been collected. The generated data was tested using a number of well-known learning techniques and the experimental results showed that the proposed methods are worthy in helping the tested learning methods perform better on this problem.

There are several potential research that are arisen from this paper. First and very important one is to evaluate the effectiveness of the proposed methods on more data sets particularly the data set was collected from the real environment as in [5]. Second, some variant of the algorithms introduced in this paper can be proposed to better reflect the attack behaviors. One of the variant could be that the new samples are generated by modifying more than one features. Third, we would like to study some probabilistic models such as Markov chain [14] to see if we can find a new and comprehensive model for generating synthetic attack data from the previously collected data. Last but not least, the efficiency of these methods will be better evaluated if they are compared to some similar methods in machine learning such as re-balancing data techniques and one-class learning [12]. In the future, we are planning to do this.

8. ACKNOWLEDGMENTS

The work in this paper was funded by The Vietnam National Foundation for Science and Technology Development (NAFOSTED), under grant number 102.01-2014.09. The Network Security Lab of Le Quy Don University provided research facilities for this study.

9. REFERENCES

[1] M. S. Abadeh, J. Habibi, Z. Barzegar, and M. Sergi. A parallel genetic local search algorithm for intrusion detection in computer networks. *Eng. Appl. of AI*, 20(8):1058–1069, 2007.

[2] C. C. Aggarwal. *Outlier Analysis*. Springer, 2013.

[3] H. B. Barlow. Unsupervised learning. *Neural Computation*, 1:295–311, 1989.

[4] F. Bergadano. Machine learning and the foundations of inductive inference. *Minds and Machines*, 3(1):31–51, 1993.

[5] V. L. Cao, V. T. Hoang, and Q. U. Nguyen. A scheme for building a dataset for intrusion detection systems. In *the 2013 Third World Congress on Information and Communication Technologies*, pages 120–132, Hanoi-Vietnam, 2013. IEEE.

[6] W.-H. Chen, S.-H. Hsu, and H.-P. Shen. Application of SVM and ANN for intrusion detection. *Computers & OR*, 32:2617–2634, 2005.

[7] Y. Chen, A. Abraham, and B. Y. 0001. Hybrid flexible neural-tree-based intrusion detection systems. *Int. J. Intell. Syst*, 22(4):337–352, 2007.

[8] N. Cristianini and J. Shawe-Taylor. *An introduction to Support Vector Machines*. Cambridge University Press, Mar. 2000.

[9] S. Das. Elements of artificial neural networks. *IEEE Transactions on Neural Networks*, 9(1):234–235, Jan. 1998.

[10] D. E. Denning. An intrusion-detection model. *IEEE Transactions on Software Engineering*, 13(2):222–232, Feb. 1987.

[11] W. Fan, M. Miller, S. Stolfo, W. Lee, and P. Chan. Using artificial anomalies to detect unknown and known network intrusions. In *Proceedings of ICDM01*, pages 123–248, 2001.

[12] S. Garcia and F. Herrera. Evolutionary undersampling for classification with imbalanced datasets: Proposals and taxonomy. *Evolutionary Computation*, 17(3):275–306, 2009.

[13] G. Giacinto, R. Perdisci, M. D. Rio, and F. Roli. Intrusion detection in computer networks by a modular ensemble of one-class classifiers. *Information Fusion*, 9(1):69–82, 2008.

[14] R. Givan, S. Leach, and T. Dean. Bounded-parameter Markov decision processes. *Artificial Intelligence*, 122(1–2):71–109, 2000.

[15] D. Heckerman. Tutorial on learning in bayesian networks. Technical Report MSR-TR-95-06, Microsoft, 1995.

[16] N. Intrator. On the combination of supervised and unsupervised learning. *Physica A*, pages 655–661, 1993.

[17] W. Lee, S. Stolfo, and K. Mok. A data mining framework for building intrusion detection models. In *Proceedings of the 1999 IEEE Symposium on Security and Privacy (SSP '99)*, pages 120–132, Washington - Brussels - Tokyo, 1999. IEEE.

[18] W. Lee and S. J. Stolfo. A framework for constructing features and models for intrusion detection systems. *ACM Trans. Inf. Syst. Secur*, 3(4):227–261, 2000.

[19] Y. Li and L. Guo. An active learning based TCM-KNN algorithm for supervised network intrusion detection. *Computers & Security*, 26(7-8):459–467, 2007.

[20] Y. Liu, K. Chen, X. Liao, and W. Zhang. A genetic clustering method for intrusion detection. *Pattern Recognition*, 37(5):927–942, 2004.

[21] T. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[22] S. Mukkamala, A. H. Sung, and A. Abraham. Intrusion detection using an ensemble of intelligent paradigms. *J. Network and Computer Applications*, 28(2):167–182, 2005.

[23] Quinlan. Learning decision tree classifiers. *CSURV: Computing Surveys*, 28, 1996.

[24] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA, 1993.

[25] K. Shafi and H. A. Abbass. Evaluation of an adaptive genetic-based signature extraction system for network intrusion detection. *Pattern Anal. Appl*, 16(4):549–566, 2013.

[26] C.-F. Tsai, Y.-F. Hsu, C.-Y. Lin, and W.-Y. Lin. Intrusion detection by machine learning: A review. *Expert Systems with Applications*, 36(10):11994–12000, 2009.

[27] V. Vapnik. *Statistical Learning Theory*. Wiley, 1998.

[28] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2005.

[29] S. X. Wu and W. Banzhaf. The use of computational intelligence in intrusion detection systems: A review. *Appl. Soft Comput*, 10(1):1–35, 2010.

[30] H. Zhang. The optimality of naive bayes. *17th International FLAIRS conference, Miami Beach, May*, pages 17–19, 2004.