Adaptive Intrusion Detection Based on Machine Learning: Feature Extraction, Classifier Construction and Sequential Pattern Prediction

Xin Xu*

Institute of Automation, College of Mechantronics Engineering and Automation,
National University of Defence Technology, Changsha, 410073, P.R.China
xuxin_mail@263.net

***Abstract*:** In recent years, intrusion detection has emerged as an important technique for network security. Due to the large volumes of security audit data as well as complex and dynamic properties of intrusion behaviors, to optimize the performance of intrusion detection systems (IDSs) becomes an important open problem. In this paper, a general framework of adaptive intrusion detection based on machine learning is presented. In the framework, three perspectives of challenging problems are explored, which include feature extraction, classifier construction and pattern prediction for sequential data. It is illustrated that the three perspectives of research challenges are mainly suitable for machine learning methods using unsupervised, supervised and reinforcement learning algorithms, respectively. Several recently developed machine learning algorithms, including a multi-class support vector machine with principal component analysis (PCA) for feature reduction and a reinforcement learning algorithm for sequential prediction, are applied and evaluated both on network-based traffic data and on host-based program behaviors. Experiments on the KDD99 intrusion detection data set and the system call data from University of New Mexico show very promising results for the machine learning approaches to adaptive intrusion detection. Some directions for future research works are also discussed.

***Keywords*:** Intrusion Detection, Machine Learning, Support Vector Machines, Reinforcement Learning

1. Introduction

In the era of information society, as computer networks and related applications become more and more popular, the potential threats to the global information infrastructure have increased a lot. To defend various cyber attacks and computer viruses, lots of computer security techniques have been studied in the last decade, which include cryptography, firewalls and intrusion detection systems (IDSs), etc. Among these techniques, intrusion detection [1-2] has been considered to be more promising for defending complex and dynamic intrusion behaviors since different behavior models or patterns can be constantly developed to detect intrusions and after successful detection of intrusions, various response techniques can be employed to stop and trace intrusion behaviors. Thus, one of the central problems for IDSs is to build effective behavior models or patterns to distinguish normal behaviors from abnormal behaviors by observing collected audit data. To solve this problem, earlier IDSs usually rely on security experts to analyze the audit data and construct intrusion detection rules manually [2]. However, since the amount of audit data, including network traffic, process execution traces and user command data, etc., increases vary fast, it has become a time-consuming, tedious and even impossible work for human experts to analyze and extract attack signatures or detection rules from dynamic, huge volumes of audit data. Furthermore, detection rules constructed by human experts are usually based on fixed features or signatures of existing attacks, so it will be very difficult for these rules to detect deformed or even completely new attacks.

Due to the above deficiencies of IDSs based on human experts, intrusion detection techniques using data mining have attracted more and more interests in recent years. Because of its inter-disciplinary nature, the advances in data mining have received contributions from many disciplines, where statistics and machine learning are the most important areas [3-6]. As an important application area of data mining, intrusion detection based on data mining algorithms, which is usually referred to as adaptive intrusion detection, aims to solve the problems of analyzing huge volumes of audit data and realizing performance optimization of detection rules. By making use of data mining algorithms, adaptive intrusion detection models can be automatically constructed based on labeled or unlabeled audit data.

Until now, to model attack behaviors or features using intrusion audit data, various association data mining algorithms [3-4], fuzzy logic models [6], and neural networks [7] have been used. The above approaches to intrusion detection are usually called misuse detection methods. The other type of intrusion detection approaches is anomaly detection, which is to model normal usage behaviors by employing data mining methods based on statistics. In anomaly detection, attacks are identified as deviations from models of normal usage.

Despite of many advances that have been achieved, existing IDSs still have some difficulties in improving their performance to meet the needs of detecting increasing types

* Corresponding author. This work is supported by the National Natural Science Foundation of China under Grant 60303012

of attacks in high-speed networks. One difficulty is to improve detection abilities for complex or new attacks without increasing false alarms. Since misuse IDSs employ signatures of known attacks, it is hard for them to detect deformed attacks, notwithstanding completely new attacks. On the other hand, although anomaly detection can detect new types of attacks by constructing models of normal behaviors, the false alarm rates in anomaly-based IDSs are usually high. How to increase the detecting ability of IDSs while maintaining low false alarms is still an open problem. Another difficulty of current IDSs is to realize real-time detection in high-speed network traffics. Since in high-speed networks, IDSs have to deal with large volumes of data in a short time, the detection rules in IDSs can not make use of lots of data features. Therefore, how to reduce the feature dimension of existing IDSs while maintaining high detection accuracy remains another challenging problem for IDS research. In addition, there still exist other difficult problems such as sequential behavior prediction in host-based IDSs.

In this paper, based on a comprehensive analysis for the current research challenges in intrusion detection, a framework for adaptive intrusion detection using machine learning techniques is presented, which includes feature extraction, classifier construction and sequential pattern prediction. Within this framework, some recently developed machine learning methods for intrusion detection are applied to the IDS problem and their performances are evaluated based on experiments on public benchmark datasets such as the KDD99 dataset. Although various hybrid approaches may be employed, it is illustrated that these three perspectives of research challenges are mainly suitable for machine learning methods using unsupervised, supervised and reinforcement learning algorithms, respectively. In contrast, in the previous adaptive IDS framework in [3], feature selection and classifier construction of IDSs were mainly tackled by traditional association data mining methods such as the Apriori algorithm.

The paper is organized as follows. In Section 2, the basic concepts and problems of intrusion detection are introduced. In Section 3, three perspectives of the challenging problems in IDSs, which are suitable for machine learning, are analyzed. In Section 4, 5 and 6, the applications of three recently developed machine learning algorithms in IDSs are presented for the purposes of classifier construction, dimension reduction, and sequential pattern prediction, respectively. Some discussions on future research work are given in Section 7.

2. Basic Problems of Intrusion Detection

The earliest intrusion detection model was proposed by Denning [1] and many research works have been devoted to the construction of effective intrusion detection models hereafter. According to the different types of audit data, IDSs can be divided into two categories, i.e., network-based IDSs and host-based IDSs. A network-based IDS monitors the contents as well as the formats of network traffic data which are usually irrelevant to the operating systems in host computers. In contrast, a host-based IDS detects possible attacks or viruses into host computers by collecting information specific to the operating systems of the target computers, which include system call traces of processes, user shell command, etc.

To realize performance optimization of IDSs as well as the task of analyzing huge volumes of audit data, lots of data mining methods for intrusion detection have been studied in the literature. Thus, how to evaluate the performance of different data mining methods becomes a critical problem in IDS research. In 1998, to compare the performance of various intrusion detection methods based on data mining, a simulated environment was set up by the MIT Lincoln Lab and nine weeks of raw TCP dump data for a local-area network (LAN) were obtained. This dataset, which is usually called DARPA98 evaluation data, has received much attention in the research community of adaptive intrusion detection. Since the raw TCP dump data can not be processed by data mining algorithms directly, a framework of feature extraction and inductive rule learning for the DARPA98 dataset was proposed by W.K. Lee, et al [3]. Later in 1999, based on the DARPA98 data and the work of W.K. Lee, the Third International Knowledge Discovery and Data Mining Tools Competition established the KDD99 benchmark data set for intrusion detection based on data mining. In the KDD99 data set, each data record corresponds to the features of a connection in the network data flow. Each connection is labeled either as normal or as an attack, with exactly one specific attack type. The data records are all labeled with one of the following five types:

- Normal: Normal connections are generated by simulated daily user behavior such as visiting web pages, downloading files, etc.
- DoS: DoS denotes the denial of service attack. A denial of service attack causes the computing power or memory of a victim machine too busy or too full to response to legitimate access. Examples of DoS attacks are Apache2, Back, Land, Mail bomb, etc.
- U2R: U2R means user to root, which is a class of attacks that a hacker begins with the access of a normal user account and then becomes a super-user by exploiting various vulnerabilities of the system. Examples are Eject, Ffbconfig, Fdformat, and Loadmodule.
- R2L: The R2L attack or remote to local attack is a class of attacks that a remote user gains access of a local account by network communication, which include Sendmail, Xlock, and Xsnoop.
- Probe: A Probe attack scans the network to gather information of computers so that vulnerabilities can be found for further attacks.

The performance criteria in KDD99 were based on the following confusion matrix:

Table 2.1 Confusion matrix of IDSs

	Normal	Probe	U2R	R2L	DoS
Normal	A11	A12	A13	A14	A15
Probe	A21	A22	A23	A24	A25
U2R	A31	A32	A33	A34	A35
R2L	A41	A42	A43	A44	A45
DoS	A51	A52	A53	A54	A55

In the above confusion matrix, the element *Aij* ($1 \le i \le 5$, $1 \le j \le 5$) denotes the number of records that belong to class *i* and were classified as class *j* by IDSs. Therefore, based on the confusion matrix, we can easily compute other performance criteria such as the detection rate of class *i*:

$$R_d(i) = \frac{A_{ii}}{\sum_{j=1}^{5} A_{ij}} \quad (1)$$

And the false alarm rate of IDSs can be computed by

$$R_f = 1 - \frac{A_{11}}{\sum_{j=1}^{5} A_{1j}} \quad (2)$$

The KDD99 dataset was established from network flow data, so it is only used for performance evaluation of network-based IDSs. For host-based intrusion detection, the audit data are usually obtained by collecting execution trajectories of processes or user commands in a host computer. As discussed in [8], host-based IDSs can be realized by observing sequences of system calls, which are related to the operating systems in the host computer. The execution trajectories of different processes form different traces of system calls. Here, each trace is defined as the list of system calls issued by a single process from the beginning of its execution to the end. A simple case of a process trace consisting 7 system calls is shown as follows:

open, read, mmap, mmap, open, read, mmap

In the construction of host-based intrusion detection model using sequences of system calls, a certain amount of normal traces as well as attack traces are collected and labeled by human experts. To detect abnormal behavior or attacks based on the system call traces, state transition models were commonly used to distinguish normal traces from abnormal traces, where the states can be defined as short sequences of system calls in a single trace. For example, if we select a sequence of 4 system calls as one state and the sliding length between sequences is 1, the state transitions corresponding to the above simple trace are:

State 1: open, read, mmap, mmap

State 2: read, mmap, mmap, open

State 3: mmap, mmap, open, read

State 4: mmap, open, read, mmap

Although some research work has been done to transform the above problem to a static pattern matching or classification problem [9], i.e., a class label is assigned to every state or short sequence, dynamic behavior models for sequential pattern prediction have been shown to be superior to static models, which has been studied and verified in [10]. Hence, the detection of sequential abnormal behaviors in host-based intrusion detection is more suitable to be regarded as a sequential pattern prediction problem, which is different from the pattern classification problem in network-based intrusion detection. Later in Section 6, we will study a recently developed learning prediction approach [24] based on reinforcement learning for host-based intrusion detection.

3. Three Perspectives of Challenges in IDSs

Although IDSs have been used as commercial products in industry, the performance of current IDSs can not satisfy the need for defending increasing number of attack types since most commercial IDSs are still based on expert rules that are manually constructed by human experts and only describe known attack signatures. In the research community, intrusion detection based on data mining has been widely studied. However, there is still much work to do to make IDSs based on data mining be applied widely in industry and completely take the place of existing IDS products using expert rules. In this section, we will analyze the technical challenges in IDSs, which are eligible for new data mining algorithms based on various machine learning methods. In the following, we will analyze three perspectives of technical challenges in IDSs based on machine learning, which are feature extraction, classifier construction and sequential pattern prediction.

To explain the three perspectives of technical challenges, a general framework for IDSs based on machine learning is presented in Figure 1.

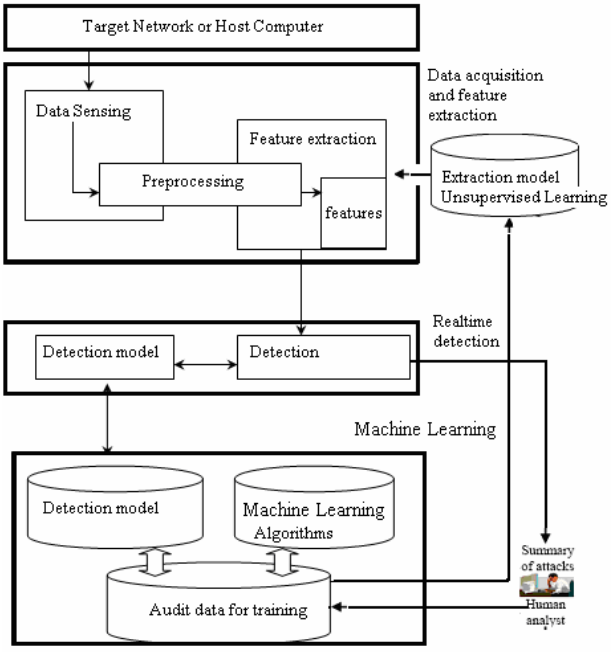

Figure.1 A framework for IDSs based on machine learning

The framework is composed of three main parts. The first one is for data acquisition and feature extraction. Data acquisition is realized by a data sensing module that observes network flow data or process execution trajectories from host computers. After pre-processing of the raw data, a feature extraction module is used to convert the raw data into feature vectors that can be processed by machine learning algorithms and an extraction model based on unsupervised learning can be employed to extract more useful features or reduce the dimensionality of the feature vectors. This process for automated feature extraction is a component of the machine learning part in the framework. In the machine learning part, audit data for training are stored in databases and they can be dynamically updated either by human analysts or by machine

learning algorithms. To automatically construct detection models from the audit data, various machine learning methods can be applied, which include unsupervised learning, supervised learning and reinforcement learning. In the following sections, we will apply some recently developed machine learning algorithms to the challenging problems in IDSs and evaluate their performance. The third part in the framework depicted in Figure.1 is for real-time detection, which is to make use of the detection models as well as the extracted feature vectors to determine whether an observed pattern or a sequence of patterns is normal or abnormal.

3.1 Feature extraction

As illustrated in Fig.1, feature extraction is the basis for high-performance intrusion detection using machine learning methods since the detection models have to be optimized based on the selection of feature spaces. If the features are improperly selected, the ultimate performance of detection models will be influenced a lot. This problem has been studied during the early work of W.K. Lee [3] and his research results lead to the benchmark dataset of KDD99, where a 41-dimensional feature vector was constructed for each network connection. The feature extraction method in KDD99 made use of various data mining techniques to identify some of the important features for detecting anomalous connections. As we will discuss later, the features employed in KDD99 can serve as the basis of further feature extraction. Here, we will briefly introduce some properties of the data and features in KDD99 data set.

In KDD99, there are 494,021 records in the 10% training data set and the number of records in the testing data set is about five million, with a 10 percent testing subset of 311028 records. The data set contains a total amount of 22 different attack types. There are 41 features for each connection record that have either discrete values or continuous values. The 41-dimensional feature can be divided into three groups. The first group of features is called basic or intrinsic features of a network connection, which include the duration, prototype, service, number of bytes from source IP addresses or from destination IP addresses, and some flags in TCP connections. The second group of features in KDD99 is composed of the content features of network connections and the third group is composed of the statistical features that are computed either by a time window or a window of certain kind of connections.

The feature selection method in the KDD99 dataset has been widely used as a standard method for network-based intrusion detection. However, in the later work of other researchers, it was found that the 41-dimensional features are not the best ones for intrusion detection and the performance of IDSs may be further improved by studying new feature extraction or dimension reduction methods [12]. In Section 4 of this paper, we will study a dimension reduction method based on principal component analysis (PCA) [11] so that the classification speed of IDSs can be improved a lot without much loss of detection precision.

3.2 Classifier construction

After performing feature extraction of network flow data, every network connection record can be denoted by a numerical feature vector and a class label can be assigned to the record, i.e.,

$$\{(\vec{x}_i, y_i)\}, \quad i = 1,2,...,N \quad y_i \in \{1,2,...,m\} \tag{3}$$

For the extracted features of audit data such as KDD99, when labels were assigned to each data record, the classifier construction problem can be solved by applying various supervised learning algorithms such as neural networks, decision trees, etc. However, the classification precision of most existing methods needs to be improved since it is very difficult to detect lots of new attacks by only training on limited audit data. Using anomaly detection strategy can detect novel attacks but the false alarm rate is usually very high since to model normal patterns very well is also hard. Thus, the classifier construction in IDSs remains another technical challenge for intrusion detection based on machine learning.

3.3 Sequential pattern prediction

As discussed in the previous sections, host-based IDSs are different from network-based IDSs in that the observed trajectories of processes or user shell commands in a host computer are sequential patterns. For example, if we use system call traces as audit data, a trajectory of system calls can be modeled as a state transition sequence of short sequences, which has been illustrated in Section 2. In the following Figure. 2, it is shown that every state is a short sequence of length 3 and different system call traces can form different state transitions, where *a*, *b*, and *c* are symbols for system calls in a host computer.

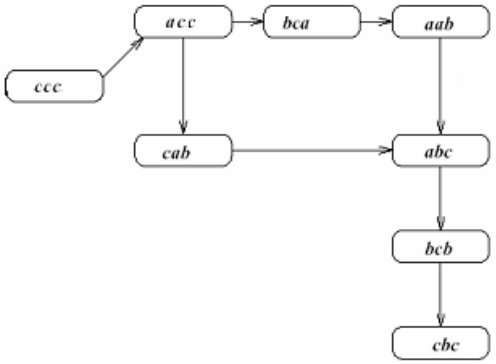

Figure 2. A sequential state transition model for host-based IDSs

Therefore, the host-based intrusion detection problem can be considered as a sequential prediction problem since it is hard to determine a single short sequence of system calls to be normal or abnormal and there are intrinsic temporal relationships between sequences. Although we can still transform the above problem to a static classification problem by mapping the whole trace of a process to a feature vector [9], it has been shown that dynamic behavior modeling methods, such as Hidden Markov Models (HMMs) [10], are more suitable for this kind of intrusion detection problem. In Section 6, we will apply a new approach for host-based intrusion detection based on reinforcement learning, where a Markov reward model is established for sequential pattern prediction and temporal difference (TD) algorithms [15] are used to realize high-precision prediction without many computational costs.

4. Multi-class Support Vector Machines for Classifier Construction

As discussed in Section 3, classifier construction is one of the central problems for network-based intrusion detection. In previous works on classifier construction of IDSs, supervised or unsupervised learning algorithms were employed but their performance was not very satisfactory due to the challenging problem of detecting novel attacks with low false alarms. In this section, we will apply multi-class Support Vector Machines (SVMs) [13] to classifier construction in IDSs and evaluate the performance of SVMs on the KDD99 dataset. Compared with the winner's performance in KDD-Cup99 [20], where a bagged boosting C5.0 classifier was used, the multi-class SVMs can obtain comparable results only by making use of a very small portion of the training data. The promising results clearly illustrate the learning efficiency and generalization ability of SVMs based on statistical learning theory.

4.1 Multi-class SVMs for intrusion detection

Based on the idea of constructing optimal hyper-planes to improve generalization abilities, SVMs were originally proposed for binary classification problems. Nevertheless, most real world pattern recognition applications are multi-class classification cases. Thus, multi-class SVM algorithms have received much attention over the last decades and several decomposition-based approaches for multi-class problems have been proposed [17-18].

The idea of decomposition-based methods is to divide a multi-class problem into multiple binary problems, i.e., to construct multiple two-class SVM classifiers and combine their classification results. There are several strategies for the implementation of multi-class SVMs using binary SVM algorithms, which include one-vs-all, one-vs-one, and error correcting output coding (ECOC) [18], etc. Among the existing decomposition approaches, the one-vs-all strategy has been regarded as a simple method with relatively low precision when compared with other multi-class SVMs. However, a recent work in [17] demonstrated that one-vs-all classifiers are also extremely powerful and can produce results that are usually at least as accurate as other methods. Therefore, in our application, we will employ the one-vs-all strategy for multi-class SVMs, where a binary SVM classifier is constructed for each partition of training data sets. For *m* classes of data, there will be *m* binary SVM classifiers to be built based on different partitions of the training data. Thus, the multi-class classification problem is decomposed into *m* subtasks of training binary SVM classifiers.

In the training of binary SVM classifiers, a hyperplane is constructed to separate two classes of samples, where a linear form of separating hyperplanes can be described as follows:

$$(\vec{w}\cdot\vec{x})+b=0 \;\; \vec{w}\in R^n, \;\; b\in R \tag{4}$$

Then, the decision function can be given by

$$f(x)=\mathrm{sgn}(\vec{w}\cdot\vec{x}+b) \tag{5}$$

Based on the structural risk minimization (SRM) principle from the statistical learning theory, the optimal linear separating hyperplane can be constructed by the following optimization problem

$$\min_{\vec{w},b}\frac{1}{2}\|\vec{w}\|^2 \tag{6}$$

subject to

$$y_i(\vec{w}\cdot\vec{x}_i+b)\geq 1, \quad i=1,2,...,N \tag{7}$$

To reduce the effects of noise and outliers in real data, soft margin techniques are usually used and the primal optimization problem becomes

$$\min_{\vec{w},b}\frac{1}{2}\|\vec{w}\|^2+C\sum_{i=1}^{N}\xi_i \tag{8}$$

subject to

$$y_i(\vec{w}\cdot\vec{x}_i+b)\geq 1-\xi_i, \quad \xi_i\geq 0, \;\; i=1,2,...,N \tag{9}$$

The Lagrangian dual of soft-margin support vector learning can be formulated as

$$\max_{\alpha}\sum_{i=1}^{N}\alpha_i-\frac{1}{2}\sum_{i,j=1}^{N}\alpha_i\alpha_j y_i y_j(\vec{x}_i\cdot\vec{x}_j) \tag{10}$$

subject to

$$0\leq\alpha_i\leq C, \;\; i=1,2,...,N \;\text{ and }\; \sum_{i=1}^{N}\alpha_i y_i=0 \tag{11}$$

An important element for the success of SVMs is the 'kernel trick', which is to transform the above linear form of support vector learning algorithms to nonlinear ones without explicitly computing the inner products in high-dimensional feature spaces. In the kernel trick, a Mercer kernel function $k(.,.)$ is employed to express the dot products in high-dimensional feature space

$$k(\vec{x}_i,\vec{x}_j)=(\vec{x}_i\cdot\vec{x}_j) \tag{12}$$

By introducing the kernel function, the dual optimization problem of SVMs for two-class soft margin classifiers can be formulated as follows

$$\max_{\alpha}\sum_{i=1}^{N}\alpha_i-\frac{1}{2}\sum_{i,j=1}^{N}\alpha_i\alpha_j y_i y_j k(\vec{x}_i\cdot\vec{x}_j) \tag{13}$$

subject to

$$0\leq\alpha_i\leq C, \;\; i=1,2,...,N \;\text{ and }\; \sum_{i=1}^{N}\alpha_i y_i=0 \tag{14}$$

To solve the above quadratic optimization problem, various decomposition-based fast algorithms have been proposed in the literature, such as SMO [19], etc. For details on the algorithmic implementation of SVMs, please refer to [19].

In our multi-class SVM classifier, the decision function of each binary SVM is

$$f_k(\vec{x})=\mathrm{sgn}(\sum_{i=1}^{N}\alpha_{ki}y_{ki}k(\vec{x}_{ki},\vec{x})+b_k) \;\; k=1,2,...,m \tag{15}$$

where *m* is the number of total classes, $f_k(\vec{x})$ is the decision function of classifier *k* and $(\vec{x}_{ki}, y_{ki})$ (k=1,2,…,m) are the corresponding training samples.

Based on the decision functions of *m* binary SVMs, a voting strategy is used to obtain the final results, i.e., the decision of each binary classification is considered to be a vote and the final decision is designated to be the class with maximum number of votes.

4.2. Performance evaluation

The performance of the multi-class SVMs for intrusion detection was evaluated on the KDD-99 dataset. In the evaluation experiments, 10 percent of the whole KDD99 training dataset and 10 percent of the testing data were both selected. To illustrate the learning efficiency of the multi-class SVMs, only a very small part (about 2%) of the 10 percent training data was used as the actual training data. Table 4.1 shows the distribution of connection types in the KDD99 10% training dataset, which has a total number of 494021 records and the record numbers of Normal, Probe, DoS, U2R and R2L connections are 97277, 4107, 391458, 52, and 1126, respectively. From the distribution of the 10 percent training set, it is shown that the numbers of different classes of data are imbalanced in this data set.

Table 4.1 Distribution of connection types in KDD99 10% training data set

Class	Number of connections
Normal	97277
Probe	4107
DoS	391458
U2R	52
R2L	1126
Total	494021

Table 4.2 Distribution of connection types in KDD99 10% testing data set

Class	Number of connections
Normal	60592
Probe	4166
DoS	237594
U2R	70
R2L	8606
Total	311028

Table 4.3 Distribution of connection types in actual training data for multi-class SVMs

Class	Number of connections
Normal	3175
Probe	1369
DoS	4349
U2R	52
R2L	563
Total	9508

Table 4.2 and Table 4.3 list the distribution of 10% testing data and the actual training data for our multi-class SVMs, respectively. In Table 4.2, it is shown that the distribution of different classes in testing data is also imbalanced but deviates from the distribution of training data. Furthermore, there are some new attacks in the 10% testing set. In our experiments, the multi-class SVMs were trained on a training set which has only 9508 records and the proportions of different classes are adjusted to solve the imbalanced data problem.

Since the multi-class SVMs were trained only on a small portion of the whole training dataset (about 2%), the computational cost in the training process is very low when compared with other data mining methods that use a large part of the whole training data set. In all the experiments, radius basis function (RBF) kernel functions are used and the width parameter is chosen as σ =0.1. After training, the multi-class SVMs were tested both on the training and testing dataset shown on Table 4.1 and 4.2, respectively.

The confusion matrix of SVMs on the whole 10% training dataset, which has a total record number of 494021, is given in Table 4.4. The detection rates of the five classes, i.e., Normal, Probe, DoS, U2R, R2L, are 99.5%, 98.7%, 94.7%, 99%, and 97.2%, respectively. And the false alarm rate is 0.5%. From the results, it can be seen that the performance of multi-class SVMs is very good on the training data set and the classifier was constructed only using a very small portion of the training data. The following Table 4.5 shows the confusion matrix of SVMs on the 10% test dataset. Since the testing data have different class distributions and new types of attacks are added, the performance of SVMs is not as good as that in the training dataset. However, the false alarm rate is relatively low (0.6%) and the detection rates of Normal, Probe, DoS, U2R and R2L are 99.4%, 81.2%, 76.7%, 21.4% and 11.2%, respectively.

Table 4.4 Confusion matrix of SVMs on 10% training set (FA: false alarm rate)

	Normal	Probe	DoS	U2R	R2L
Normal	94758	156	0	51	312
Probe	49	4055	0	2	1
DoS	101	20466	370864	0	27
U2R	1	0	0	51	0
R2L	30	0	0	1	1095
FA	0.5%				

Table 4.5 Confusion matrix of SVMs in 10% test set (FA: false alarm rate)

	Normal	Probe	DoS	U2R	R2L
Normal	60220	221	52	48	51
Probe	699	3382	82	0	3
DoS	14189	41165	182228	1	11
U2R	45	0	0	15	10
R2L	7431	186	0	28	961
FA	0.6%				

Table 4.6 shows the confusion matrix of KDD99 winner. The false alarm rate is 0.5% and the detection rates of Normal, Probe, DoS, U2R and R2L connections are 99.5%, 83.3%, 97.1%, 13.2%, and 8.4%, respectively. Although the performance of our multi-class SVMs has not been optimized by increasing the number of training data, the detection rates on class Normal, Probe are comparable to the KDD99 winner [20] (99.4% vs. 99.5%, 83.3% vs. 81.2%) and the detection rates of U2R and R2L are better than the KDD99 winner (21.4% vs. 13.2%, 11.2% vs. 8.4%). The detection rate of DoS is not satisfactory for the multi-class SVMs and this can be improved by selecting more training data of DoS connections since we only use 4349 DoS connections in training while the KDD99 winner used about 400000 training data of DoS attacks.

Table 4.6 Confusion matrix of KDD99 Winner in 10% test set (FA: false alarm rate)

	Normal	Probe	DoS	U2R	R2L
Normal	60262	243	78	4	6
Probe	511	3471	184	0	0
DoS	5299	1328	223226	0	0
U2R	168	20	0	30	10
R2L	14527	294	0	8	1360
FA	0.5%				

The above evaluation results clearly demonstrate that the learning efficiency and generalization ability of multi-class SVMs are very promising for high-performance IDSs and the performance can be further improved by applying feature selection techniques, such as the PCA-based method reported later in this paper, or by increasing the number of training audit data. Moreover, new classification learning algorithms can also be developed to improve the performance of IDSs based on machine learning.

5. Dimension Reduction using PCA

In neural network and statistics studies, PCA is one of the most fundamental tools of dimensionality reduction for extracting effective features from high-dimensional vectors of input data. In the following, we will study the application of PCA to dimension reduction of network connection data.

As discussed in Section 2, based on the feature extraction process suggested by W.K. Lee [3], the network data records in KDD99 can be denoted as

$$x_t = [x_{t1}, x_{t2}, ..., x_{tn}]^T \quad (t=1,2,\ldots, N),\ n=41 \tag{16}$$

Let

$$\mu = \frac{1}{N}\sum_{t=1}^{N} x_t \tag{17}$$

Then, the covariance matrix of data vectors is

$$C = \frac{1}{N}\sum_{t=1}^{N}(x_t - \mu)(x_t - \mu)^T \tag{18}$$

The principal components are computed by solving the eigenvalue problem of the covariance matrix C:

$$Cv_i = \lambda_i v_i \tag{19}$$

where λ_i $(i = 1,2,...,n)$ are the eigenvalues and $v_i (i = 1,2,...,n)$ are the corresponding eigenvectors.

To represent network data records with low dimensional vectors, we only need to compute the first *m* eigenvectors which correspond to the *m* largest eigenvalues.

Let

$$\Phi = [v_1, v_2, ..., v_m], \quad \Lambda = \mathrm{diag}[\lambda_1, \lambda_2, ..., \lambda_m] \tag{20}$$

Then we have

$$C\Phi = \Phi\Lambda \tag{21}$$

In PCA, a parameter ν can be introduced to denote the approximation precision of the *m* largest eigenvectors so that the following relation holds.

$$\sum_{i=1}^{m}\lambda_i \Big/ \sum_{i=1}^{n}\lambda_i \geq \nu \tag{22}$$

Given a precision parameter ν, we can select the number of eigenvectors based on (21) and (22), and the low-dimensional feature vector of a new input data *x* can be determined as follows

$$x_f = \Phi^T x \tag{23}$$

To illustrate the effectiveness of the PCA-based dimension reduction, the proposed method was combined with the multi-class SVMs for classifier construction. The combined method was applied in the KDD99 dataset to demonstrate its detection accuracy and enhancement in processing speed. In the experiments, a subset of KDD99 data was selected and was partitioned to a training data set with 9321 records and a test set with 15705 records. The multi-class SVMs with PCA used a 12-dimensional vector for each connection record, and for comparison, multi-class SVMs using the original data dimension of 41 are also tested on the same data set.

In the experiments, it was found that the accuracy of the proposed SVM+PCA method is fairly good except that the results of class 'R2L' are not very satisfactory. The reason may be that the amount of U2R data is very small in the training data so that it will cause some information loss when dimension reduction is performed using PCA. However, this problem may be solved by collecting more training data of U2R attacks. Although the detection accuracies of SVMs without PCA are slightly better, SVMs with PCA will benefit from improved training and testing speed, which is important for high-speed network applications. Table 5.1 shows the comparisons of training and testing speed of SVMs with and without PCA. It is clear that the proposed PCA+SVMs classifier is approximately 5 times faster in training and 2 times faster in testing than conventional SVMs without PCA.

Table 5.1. Processing speed comparison

Classifiers	Training time (s)	Testing time (s)
SVM (41-dimensional feature)	151.9	30.4
SVM (Using PCA for feature extraction, 12 dimensions)	33.3	14.4

6. Temporal Difference Learning Prediction for Sequential Behaviors

As discussed in Section 2, a state transition model can be introduced for host-based intrusion detection using sequences of system calls. By selecting short sequences of system calls as states, a single trace can be regarded as a trajectory of an absorbing Markov chain. Since complete traces generated by computer programs can be labeled as normal or abnormal, we can design a reward function for each trace, where normal traces have a terminal reward of –1 and abnormal traces have a terminal reward of +1 and the reward for every intermediate state transition is 0. The value functions of the corresponding Markov chain are defined as follows:

$$V(i) = E\{\sum_{t=0}^{\infty}\gamma^t r_t \big| x_0 = i\} \tag{24}$$

As studied in our previous work [24], the value function of

a state will give a prediction probability of the underlying trace to be normal or abnormal. If we get accurate value function estimations, we can determine a trace to be normal or abnormal by comparing the value function with a predefined threshold. Thus, the host-based intrusion detection problem can be transformed to a value function prediction problem of Markov reward processes, where little *a priori* information on the state transition model is required but the data of the state transition processes can be observed.

Learning prediction of the value functions for Markov reward processes without any prior model is a central problem in reinforcement learning (RL). In RL, learning prediction is different from that in supervised learning. As pointed out by Sutton [15], the prediction problems in supervised learning are single-step prediction problems while those in reinforcement learning are multi-step prediction problems. To solve multi-step prediction problems, a learning system must predict outcomes that depend on a future sequence of observations and decisions. Thus, the theory and algorithms of multi-step learning prediction in RL have received much attention and lots of research work has been carried out [21-23].

Among the proposed multi-step learning prediction methods, temporal-difference (TD) learning [21] is one of the most popular methods. Some recent results include linear TD(λ) and LS-TD(λ) can be found in [22] and [23].

In the TD(λ) algorithm, there are two basic mechanisms which are the temporal difference and the eligibility trace, respectively. Temporal differences are defined as the differences between two successive estimations and have the following form.

$$\delta_t = r_t + \gamma \widetilde{V}_t(x_{t+1}) - \widetilde{V}_t(x_t) \tag{25}$$

where x_{t+1} is the successive state of x_t, $\widetilde{V}(x)$ denotes the estimate of the value function $V(x)$ and r_t is the reward received after the state transition from x_t to x_{t+1}.

Since the state space of a Markov chain is usually large or infinite in practice, function approximators are commonly used to approximate the value functions, where TD(λ) algorithms with linear function approximators are the most popular and well-studied ones [23]. In our implementation of TD(λ), a linear basis function is chosen as follows.

$$\phi(x) = (\phi_1(x), \phi_2(x), ..., \phi_n(x))^T \tag{26}$$

The estimated value function can be denoted as

$$\widetilde{V}_t(x) = \phi^T(x) W_t \tag{27}$$

where $W_t = (w_1, w_2, \ldots, w_n)^T$ is the weight vector.

The corresponding incremental weight update rule is

$$W_{t+1} = W_t + \alpha_t (r_t + \gamma \phi^T(x_{t+1}) W_t - \phi^T(x_t) W_t) \vec{z}_{t+1} \tag{28}$$

where the eligibility trace vector is defined as

$$\vec{z}_t(s) = (z_{1t}(s), z_{2t}(s), ..., z_{nt}(s))^T \tag{29}$$

$$\vec{z}_{t+1} = \gamma \lambda \vec{z}_t + \phi(x_t) \tag{30}$$

To apply the above TD learning algorithms in host-based intrusion detection using sequences of system calls, the traces from a host computer are divided into two classes, i.e., normal traces and attack traces. A reward function discussed above is introduced so that the trace data are transformed to the sample data of a Markov reward process. Then, the linear TD learning algorithm is employed to perform learning prediction of the Markov reward process so that the value functions are predicted. When the model training for learning prediction is completed, a value function prediction model can be constructed, which can be used to realize online detection.

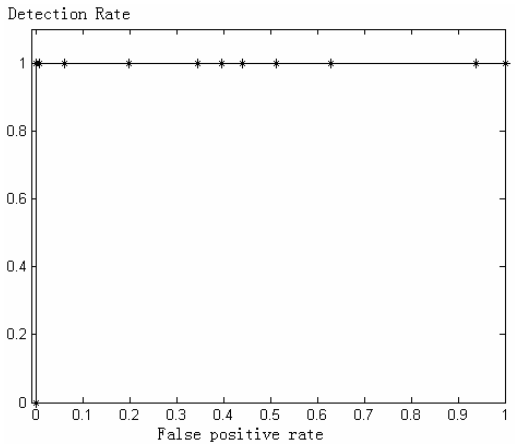

Figure. 3 ROC curves of TD learning prediction for the *lpr* data set

During the online detection process, state features are extracted from the input trace data and the vale function prediction model is used to compute value functions of the states. Then the normal or abnormal properties of the trace can be determined by the state value function and a pre-selected or optimized threshold V_0.

The performance of the RL method for intrusion detection was evaluated on the *lpr* trace data in SunOS operating systems, which can be downloaded at the website: http://www.cs.unm.edu/~immsec/dataset.html. The data for *lpr* were collected at the MIT AI laboratory environment by tracing *lpr* programs running on 77 different hosts, each running SunOS, for two weeks, to obtain traces of a total of 2766 normal print jobs. A single *lprcp* symbolic link intrusion that consists of 1001 print jobs was also obtained.

To employ the RL-based method for constructing intrusion detection models, we use only 10 normal traces and 20 abnormal traces for training. All the other traces are used as the test data. Every trace is regarded as a sample trajectory of an absorbing Markov reward process. The states of the Markov process are selected as short sequences of system calls with length 6 and the sliding length is 1. After training, the predicted value function is used to distinguish normal traces from abnormal ones by selecting a threshold value. The above Figure 3 depicts the ROC curves obtained from the performance evaluation of TD learning prediction on the testing data of MIIT *live lpr*, where different thresholds were selected and the corresponding detection rates and false alarm rates were computed. From the results, it is clearly shown that the TD learning prediction has good performance for sequential behavior prediction in host-based IDSs.

7. Discussions and Future Work

In recent years, research on data mining and machine learning for intrusion detection has received much attention

not only in the computer security community, but also in the computational intelligence community. The reason is that as an important technique for dynamic defense of computer systems, future intrusion detection systems need to meet the following two requirements. One is that huge volumes of audit data must be analyzed in order to construct new detection rules for increasing number of novel attacks. The second is that due to the increasing speed of network traffic and the dynamic and complex properties of attack behaviors, the performance of current IDSs has to be improved to meet the needs both on detection speed and detection accuracy. The two critical requirements make intrusion detection be an important application area that is eligible for machine learning.

In this paper, we analyze the three perspectives of the technical challenges for intrusion detection based on machine learning, which are feature extraction, classifier construction, and sequential pattern prediction. The comprehensive analysis in this paper can be viewed as an extension from the previous adaptive IDS framework [3] based on traditional association data mining methods to intelligent IDSs based on general machine learning algorithms. To solve the three challenging problems, we study and evaluate three new technical solutions using machine learning methods, which include PCA for feature reduction, multi-class SVMs for classifier construction and a TD learning prediction method for host-based intrusion detection. These three kinds of machine learning methods for intrusion detection make use of unsupervised learning, supervised learning, and reinforcement learning algorithms, respectively. Using various benchmark dataset for intrusion detection, such as the KDD99 data and the MIT *lpr* system call data, it is demonstrated that the machine learning methods studied in this paper are very promising to solve the three perspectives of technical challenges in future IDSs. Although further work needs to be done to improve the performance of IDSs and apply machine learning techniques in real commercial IDSs, the machine learning methods studied in this paper, i.e., PCA, multi-class SVMs, and TD prediction learning can form an important technical foundation for future work.

References

[1] D.Denning. “An intrusion-detection model”, *IEEE Transactions on Software Engineering*, 13(2), pp. 222-232, 1987.

[2] M. M. Sebring, E. Shellhouse, M. E. Hanna, and R. Alan Whitehurst. “Expert systems in intrusion detection: A case study”, *In Proceedings of the 11th National Computer Security Conference*, Baltimore, Maryland, October, pp.74-81, 1988.

[3] W.K. Lee, S.J.Stolfo. “A data mining framework for building intrusion detection model”, In: Gong L., Reiter M.K. (eds.): *Proceedings of the IEEE Symposium on Security and Privacy*. Oakland, CA: IEEE Computer Society Press, pp.120~132, 1999.

[4] W.K. Lee, et al., "Mining audit data to build intrusion detection models", In *Proc. Int. Conf. Knowledge Discovery and Data Mining (KDD'98)*, pp.66-72, 1998.

[5] N. B.Amor, S.Benferhat, and Z. Elouedi. “Naive Bayes vs decision trees in intrusion detection systems”, In *Proc. 2004 ACM Symp. on Applied Computing*. pp. 420-424, 2004.

[6] J. Luo, S. M.Bridges. “Mining fuzzy association rules and fuzzy frequency episodes for intrusion detection”, *International Journal of Intelligent Systems*, pp. 687-703, 2000.

[7] J. Ryan, M-J. Lin, R. Miikkulainen. “Intrusion detection with neural networks”, In *Proceedings of AAAI-97 Workshop on AI Approaches to Fraud Detection and Risk Management*, AAAI Press, pp. 72-77, 1997.

[8] S. Hofmeyr et al. “Intrusion detection using sequences of systems call”, *Journal of Computer Security*, 6, pp. 151-180, 1998.

[9] Y.H. Liao, V. Rao Vemuri. “Using text categorization techniques for intrusion detection”, In *Proceedings of the 11th USENIX Security Symposium*, August, pp.51-59, 2002.

[10] D.Y. Yeung, Y.X. Ding. “Host-based intrusion detection using dynamic and static behavioral models”, *Pattern Recognition*, 36, pp.229 – 243, 2003.

[11] I. T.Jolliffe. *Principal component analysis.* Springer. 2nd edition. 2002.

[12] X. Xu, X.N. Wang. “Adaptive network intrusion detection method based on PCA and support vector machines”, *Lecture Notes in Artificial Intelligence*, ADMA 2005, LNAI 3584, pp. 696 – 703, 2005.

[13] T. J.Hastie, R. J.Tibshirani, and J. H.Friedman. *The elements of statistical learning: Data mining, inference, and prediction*, Springer-Verlag, 2001.

[14] T.Lane, C.Brodley. “Temporal sequence learning and data reduction for anomaly detection”, *ACM Transactions on Information and System Security*, 2(3) pp.295–331, 1999.

[15] R.Sutton. “Learning to predict by the method of temporal differences”, *Machine Learning*, 3(1), pp. 9-44, 1988.

[16] C.-J.Lin. “Formulations of support vector machines: a note from an optimization point of view”, *Neural Computation*, 13(2), pp. 307-317, 2001

[17] R.Rifkin, A.Klautau. “In defense of one-vs-all classification”, *Journal of Machine Learning Research*, 5, pp.143-151, 2004.

[18] T. G.Dietterich, G.Bakiri. “Solving multiclass learning problems via error-correcting output codes”, *Journal of Artificial Intelligence Research*, 2, pp. 263-286, 1995.

[19] J.Platt. “Fast training of support vector machines using sequential minimal optimization”, In: *Advances in Kernel Methods—Support Vector Learning*, B.Scholkopf, C.J.C. Burges, and A.J.Smola, (eds.), Cambridge, MIT Press. pp. 185-208, 1999.

[20] B. Pfahringer. “Winning the KDD99 Classification Cup: Bagged Boosting”, *SIGKDD Explorations,* 1(2), pp. 65-66, 2000.

[21] J. A.Boyan. “Technical Update: Least-squares temporal difference learning”, *Machine Learning*, 49, pp.233 -246, 2002.

[22] X.Xu, H.G.He, D.W.Hu. “Efficient reinforcement learning using recursive least-squares methods.” *Journal of Artificial Intelligence Research*, 16, pp. 259-292, 2002.

[23] J.N.Tsitsiklis, B.V.Roy. "An analysis of temporal difference learning with function approximation". *IEEE Transactions on Automatic Control*. 42(5) pp. 674-690, 1997.

[24] X. Xu, T. Xie. "A reinforcement learning approach for host-based intrusion detection using sequences of system calls", In: *Proceedings of International Conference on Intelligent Computing*. 2005, *Lecture Notes in Computer Science*, LNCS 3644, pp.995 –1003, 2005.

Author Biographies

Xin Xu. Dr. Xin Xu received the Bachelor degree in Electrical Engineering from Department of Automatic Control, National University of Defense Technology (NUDT), P.R.China, in 1996. In 2002, he obtained the PhD degree in Electrical Engineering from College of Mechantronics Engineering and Automation (CMEA), NUDT, P.R.China. From 2003 to 2005, he was a post-doctor fellow in School of Computer, NUDT. Now he is an associate professor at Institute of Automation, CMEA, NUDT, P.R.China. His research interests include reinforcement learning, data mining, intelligent control, autonomic computing and computer security. Until now, he has published more than 30 papers on international journals and conferences, which include *Journal of Artificial Intelligence Research*, *International Journal of Information Technology*, etc. In 2003 and 2004, he received the Youth Science Foundation from the National Natural Science Foundation of China (NSFC) and the distinguished doctoral dissertation award from Hunan Province of China, respectively. In addition to being a grant reviewer of NSFC, he also served as paper reviewers for many international journals and conference.