A Hybrid Framework for Building an Efficient Incremental Intrusion Detection System

Amin Rasoulifard[i] and Abbas Ghaemi Bafghi[ii*]

Received 20 Jun 2012; received in revised 08 Jul 2012; accepted 05 Oct 2012

ABSTRACT

In this paper, a boosting-based incremental hybrid intrusion detection system is introduced. This system combines incremental misuse detection and incremental anomaly detection. We use boosting ensemble of weak classifiers to implement misuse intrusion detection system. It can identify new classes types of intrusions that do not exist in the training dataset for incremental misuse detection. As the framework has low computational complexity, it is suitable for real-time or on-line learning. We use incremental centroid-based "on-line k-Mean" clustering algorithm to implement anomaly detection system. Experimental evaluations on KDD Cup dataset have shown that the proposed framework has high clustering quality, relatively low computational complexity and fast convergence.

KEYWORDS

Intrusion Detection System, Neural Network, Incremental Learning, Boosting Ensemble Learning, Clustering, Weak Classifiers, Weak Learner

1. INTRODUCTION

Ensemble learning algorithms have become extremely popular over the last several years since these algorithms generate multiple *base* models using traditional machine learning algorithms and combine them into an *ensemble* model and have often demonstrated significantly better performance than single models. Boosting is an ensemble learning algorithm that generates a sequence of base models. Boosting maintains a probability distribution over the training set. Each base model is generated by calling the base model learning algorithm with the training set weighted by the current probability distribution. Then, the base model is tested on the training set, the weights of those examples that are misclassified by ensemble are not changed and the weights of the others are decreased accordingly. This new probability distribution and training set is used to generate the next base model. Intuitively, boosting increases the weights of previously misclassified examples thereby will focus with more attention on these hard-to-learn examples and subsequent base models correct the mistakes of the previous models [21].

Intrusion detection techniques may be categorized into misuse detection and anomaly detection [9]. Misuse detection systems use patterns of well-known attacks or weak spots of the system to identify intrusions [8], [9], [10], [15], [18], [19], [20], [36], [35], and [37]. They have high detection rate, low false positive rate and low computational complexity but the main shortcoming of such systems is the necessity of hand-coding of known intrusion patterns and their inability to detect any future (unknown) intrusions. On the other hand, anomaly detection systems establish normal user behavior patterns and try to determine whether deviations from the established normal profiles can be flagged as intrusions [3], [6], [13], [14], [24], [30] and [38]. The main advantage of anomaly detection systems is their detection of the new types of unknown intrusions. To improve the productivity of misuse and anomaly detection, several hybrid intrusion detection systems have been proposed [1], [2], [5], [11], [17], [22], [23], [29], [31], [33] and [39]. These frameworks combine misuse and anomaly detection to use the benefits of misuse detection (high detection rate, low false positive rate and low complexity) together with those of the anomaly detection (detection of the new type of unknown intrusions).

In recent years, the continual emergence of new attacking methods has caused great loss to the whole society. So, the advantage of detecting future attacks has specially led to an increasing interest in incremental learning techniques. The traditional methods commonly build a static intrusion detection model on the prior training dataset, and then utilize this model to predict on new network behavior data [6], [8], [9], [10], [13], [14], [15], [18], [19], [20], [24], [36], [37], and [38]. However, the network behavior models change continually along with detecting and analyzing process. Thus, the initially

[i] A. Rasoulifard is with the Data and Communication Security Research Laboratory, Department of Computer Engineering, Faculty of Engineering, Ferdowsi University of Mashhad, Mashhad, Iran, Iran. (email:am_ra84@stu-mail.um.ac.ir)

[ii*] Corresponding Author, A. Ghaemi B. is with the Data and Communication Security Research Laboratory, Department of Computer Engineering, Faculty of Engineering, Ferdowsi University of Mashhad, Mashhad, Iran, Iran. (email: ghaemib@um.ac.ir)

learnt intrusion detection model can not adapt to the new network behavior pattern, which causes an increase in the false positive rate and decreases the detection precision of the system.

In this paper, an incremental hybrid intrusion detection framework is introduced. This framework has high detection rate, low computational complexity, with the ability of detection of new unknown attacks and continually adapt the model to cope with new network behaviors. In addition, when intrusion detection dataset is so large that the whole dataset can not be loaded into the main memory, the original dataset can be partitioned into several subsets and the detection model is dynamically modified according to each of them.

The rest of the paper is organized as follows. Section 2 discusses background, sections 3, 4 and 5 present the proposed system architecture and its elements and KDD Cup 99 Dataset, respectively. Parameter analysis is discussed in section 6 and computational complexity is presented in section 7. Finally, conclusion and future work are presented in section 8.

2. BACKGROUND

A. Boosting works

In [10], a network based intrusion detection system is proposed using classical adaboost algorithm. The adaboost algorithm works on the instances of data based on the initial weight assigned to them. The classical adaboost is a binary classification algorithm and the initial weight of instances equals 1/n where n is number of instances. In this paper, the authors changed the objective function of a classical adaboost to adjust the tradeoff between FPR[3] and DR[4].

In [36] and [37], a multi class SLIPPER system is proposed for intrusion detection system to implement the benefit of boosting-based learning algorithms. The key idea is the use of available binary SLIPPER as a basic module, which is a rule learner based on confidence-rated boosting. It creates the ensemble of rules through boosting a weak learner. Unlike some other conventional rule learners, the covered examples are not removed from the training set. They are given lower weights in the further boosting. The rule sets will abstain on examples that are not covered by a rule, thus the label assigned to an instance depends only on the rules that "fire" on that instance.

In [25] and [26], we proposed an incremental misuse intrusion detection system using Learn++ algorithm, which uses multi layer perceptron to generate the ensemble of weak classifiers. The architecture has the ability to learn new unseen intrusions which belong to the new classes of intrusion types.

B. Hybrid system works

There are three ways to combine misuse and anomaly detection: first, using anomaly detection at before misuse detection [2], [17] and [22], second, using misuse and anomaly detection in parallel [1], [5], [31] and [33] and last, using misuse detection before anomaly detection [11] and [39].

ADAM[5] is a hybrid on-line intrusion detection system which uses association rules for detecting intrusions [2]. This framework includes two phases: training phase and detection phase. In the training phase, the dataset without any class of intrusions is applied to the model and the profile of normal activities is constituted as the set of association rules pattern. In the detection phase, ADAM uses sliding window on-line algorithm to find frequent patterns in the last D connections and compares them with those stored in the normal profile then discards those patterns which are similar to the normal profile. With the rest, ADAM uses a classifier that has been previously trained to classify the suspicious data as known attack types and unknown type.

NIDES[6] is a hybrid system [1] emerged from the rule-based misuse detection and the anomaly detection that used statistical approaches. This framework employs misuse detection and anomaly detection in parallel for detecting intrusions.

The random forest algorithm was used for the hybrid intrusion detection system in [39]. It used ensemble of classification tree for misuse detection and proximities to find anomaly intrusions. Similar to ADAM it has two phases: on-line and off-line. In on-line phase the classification trees were being used to generate the pattern of known intrusions and in the off-line phase, system detects unknown intrusions and builds the pattern of known intrusions and adds them to the database of known intrusion patterns.

FLIPS is the framework which has used hybrid approach for intrusion prevention systems [17]. The core of this framework is an anomaly-based classifier that incorporates feedback form environment to tune its model and generate the signatures of known malicious activities. It has used PayL[32] as an anomaly detection component and the misuse detection component of this framework is a signature-based intrusion detection system.

In [22], a hierarchical layering distributed intrusion detection system and response is proposed. In this paper, the analysis tool (anomaly detection) is used to generate malicious activities, and then misuse detection is applied on these activities for integrating to network computing environments. In [31], a serial combination of misuse detection and anomaly detection is proposed. They proposed a set model to formally describe anomaly and misuse intrusion detection results. They used some

3- False Positive Rate
4- Detection Rate
5- Audit Data Analysis and Mining
6-Next Generate Intrusion Expert System

notations for each intrusion detection result, and then obtained the final results based on the conflicts among them. This framework used misuse detection and anomaly detection in parallel.

In [5], a novel intrusion detection system architecture utilizing both anomaly and misuse detection approaches is proposed. This hybrid intrusion detection system consists of an anomaly detection module and a misuse detection module. A decision support system combines the results of these two detection models. They used Self-Organized Map structure to model normal behavior and used J.48 decision tree for misuse detection component. In [11], a hybrid intrusion detection system was proposed which combined the advantages of low false-positive rate of signature-based intrusion detection systems and the ability of anomaly detection systems to detect novel attacks. This framework uses anomaly detection to detect anomaly activities and uses a weighted signature generation scheme as misuse detection component. In [33] a multi-level hybrid intrusion detection system was proposed that uses a combination of tree classifiers and clustering algorithms to detect intrusions. In [25] and [26] we proposed an incremental hybrid intrusion detection system based on ensemble of weak classifiers.

3. THE ARCHITECTURE OF PROPOSED SYSTEM

Boosting ensemble learning focuses on minority or hard-to-learn instances. Usually, the majority of instances in intrusion detection systems are normal instances and the few of instances are new intrusions. So we introduce incremental hybrid intrusion detection system based on boosting ensemble of weak classifiers to focus on detecting and learning of new intrusions. Intrusion detection systems using ensemble of weak classifiers generally have lower computational complexity comparing to other frameworks which use strong classifiers. This property is very attractive and promising in intrusion detection systems as classifiers should be retained in the short periods in practice. The framework of the proposed hybrid intrusion detection system is shown in Figure 1.

This framework includes two phases: on-line and off-line. In the on-line phase, we used incremental misuse intrusion detection which works based on the ensemble of weak classifiers [7]. In the off-line phase, we used a centroid-based *k-mean* clustering algorithm [40]. The new intrusions identified by anomaly detection component should be applied to the misuse intrusion detection component in the next learning phase. In order to implement multi-class intrusion detection system, we should determine the class types of new intrusions. For this purpose, we introduced a *self-labeled intrusions classifier* based on unsupervised clustering. This component gets new intrusion items and classifies them to four class types of intrusion. The output is a new dataset that will be added to the on-line phase.

4. ARCHITECTURE ELEMENTS

A. Weak Classifier

Weak classifiers obtain 50 percent classification accuracy on training data and have low computational complexity [7] and [21]. Figure 2 shows the framework of the weak classifier. We used a single hidden layer fully connected Multi Layer Perceptron (MLP) with 41 hidden layer nodes and 4 nodes in the output layer for the weak classifier used to generate individual hypotheses.

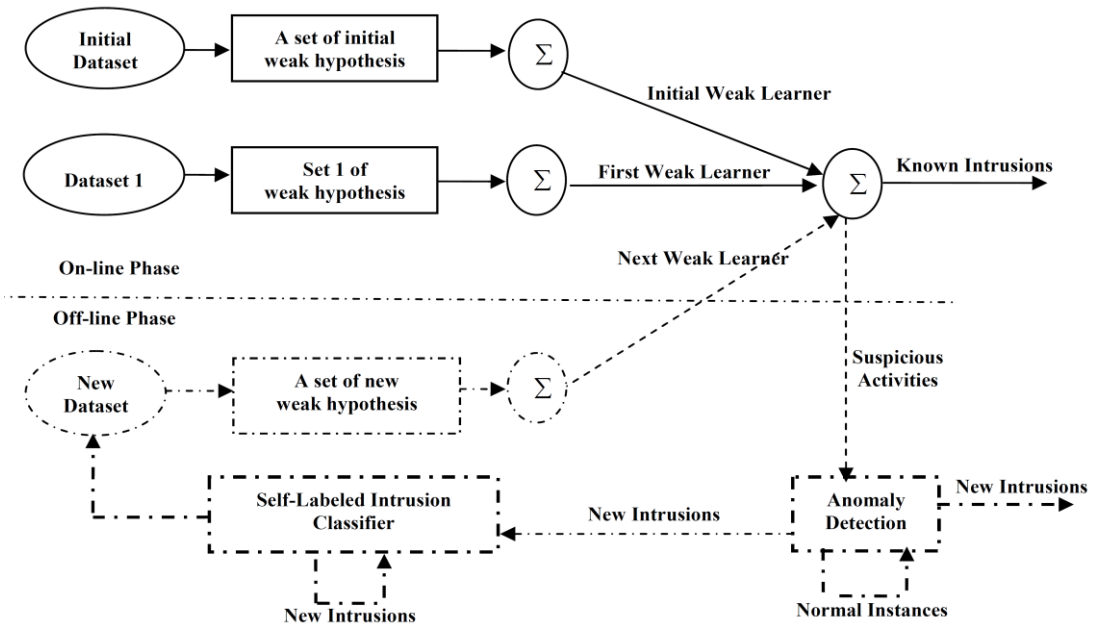

Figure 1: Incremental hybrid intrusion detection system.

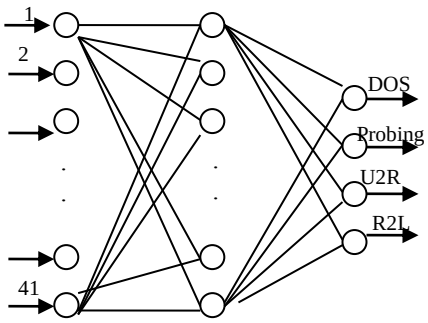

Figure 2: Weak classifier.

We chose 41 hidden nodes that is equal to the number of input variables (attributes) for each connection record of KDD cup. Some attributes (e.g. flag, protocol and service type) were nonnumeric, so we preprocess them to obtain a numerical field. For example, the protocol type of UDP, TCP and ICMP were coded by 1, 2 and 3 respectively. The four nodes in the output layer correspond to the four class types of intrusion. We note that any neural network can be turned into weak learning algorithms by selecting a high error goal with respect to the complexity of problems.

B. Weak Learner

The weak learner as shown in Figure 3, is composed of a set of weak hypotheses with generated by the boosting algorithm [21], and combined by the weighted majority-voting algorithm [16].

C. Weighted Majority Voting (WMV)

Initially a positive weight is associated with each pool hypotheses. All weights are initiated to 1 unless specified otherwise. WM [16] forms its prediction by comparing the total weight of the hypotheses of the pool that predicting false to the total weight of the hypotheses predicting true. It predicts according to the larger total weight. When it makes a mistake, the weights of disagreed hypotheses are multiplied by the fixed value of β such that $0 \le \beta \prec 1$. We use $\sum$ to show the weighted majority-voting algorithm.

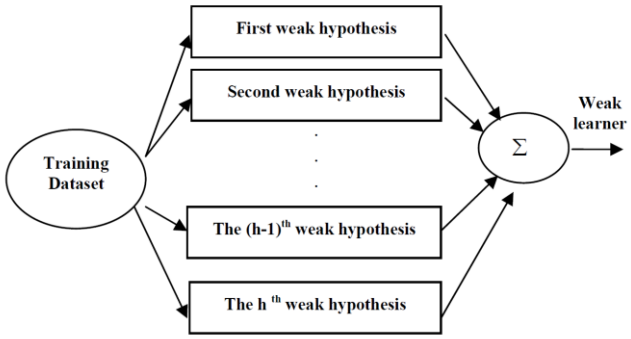

Figure 3: Weak Learner.

D. Misuse Intrusion Detection System

Misuse intrusion detection system is based on ensemble of weak classifiers [7]. In this paper, we implement two misuse intrusion detection systems based on boosting ensemble of weak classifiers and compare them in different situations to investigate on incremental learning behavior of them. These frameworks can learn the new class type of intrusion that did not exist in the previous training datasets.

E. Anomaly Detection

Anomaly detection amounts to training models for normal behavior and then classifying any network behavior that significantly deviates from the known normal patterns as intrusions. Clustering algorithms have recently gained attention in intrusion detection systems due to their advantages to find new attacks which are not seen before. We use incremental clustering algorithm in the proposed hybrid system for learning new unseen intrusions. We use *k-mean* algorithm [40]. It has low complexity, fast convergence and is suitable for incremental learning.

5. KDD CUP 99 DATASET

The KDD cup 1999 intrusion detection dataset [12] and [4] was used in our experiments. The 1998 DARPA Intrusion Detection Evaluation program by MIT Lincoln Labs (MIT Lincoln Laboratory) has prepared this dataset. Lincoln labs acquired nine weeks for raw TCP dump data. The raw data were processed into connection records, which consist of about 5 million connection records. The data set contains 24 attack types. These attacks fall into four main categories of denial of service, remote to user, user to root and probing.

Denial of Service (DoS): In this type of attack an attacker makes some computing or memory resources too busy or too full to handle legitimate requests and denies legitimate users access to a machine. Some examples are apache2, Back, Land, Mail bomb, SYN Flood, Ping of death, Process table, Smurf and Teardrop.

Remote to User (R2L): In this type of attack an attacker, who does not have an account on a remote machine, sends some packets to that machine over a network and exploits some vulnerabilities to gain local access as a user of that machine. Some examples are Dictionary, FTP write, Guest, IMAP, Named, Phf, Send mail and Xlock.

User to Root (U2R): In this type of attacks an attacker starts out with access to a normal user account on the system and exploit system vulnerabilities to gain root access to the system. Some examples are Eject, Load Module, Ps, Xterm, Perl and Fdformat.

Probing: In this type of attacks, an attacker scans a network of computers to gather information or find known vulnerabilities. An attacker with a map of machines and services that are available on a network can use this information to look for exploits. Some examples are Ispsweep, Mscan, Saint, Satan and Nmap.

The data set has 41 attributes for each connection record as well as a class label. R2L and U2R attacks do not have any sequential patterns like DoS and Probe because the former have the attacks embedded in the data packets whereas the later have many connections in a short time. Therefore, some features that look for suspicious behavior in the data packets (like number of failed logins) are constructed and called content features.

An original sample consists of about 4898431 records obtained from the UCI machine learning repository[7] was used in our study as the training set and the entire labeled test set was used for the testing set. The labeled test dataset includes 311029 records with different distribution from the training set. The distribution of normal and attack types of the connection records in these subsets have been summarized in Tables 1, 2 and 3.

TABLE 1
THE DISTRIBUTION OF ORIGINAL TRAINING DATASET OF KDD CUP 99 DATASET

Class	Number of sample	Percent
Normal	972780	19.85
Dos	3883370	79.27
U2R	52	0.001
R2L	1127	0.023
Probe	41102	0.83
Total	4898431	100

Table 2
The distribution of training 10% dataset of KDD Cup 99 dataset.

Class	Number of sample	Percent
Normal	97278	19.69
Dos	391458	79.24
U2R	52	0.01
R2L	1126	0.023
Probe	4107	0.83
Total	494021	100

Table 3
The distribution of testing dataset of KDD Cup 99 dataset.

Class	Number of sample	Percent
Normal	60593	19.48
Dos	223298	71.19
U2R	39	0.01
R2L	5993	1.92
Probe	2377	0.76
Novel	18729	6.02
Total	311029	100

6. PARAMETER ANALYSIS

To validate the effectiveness of incremental hybrid intrusion detection using ensemble of weak classifiers, the following experiments are done.

A. Weak Classifier

The 10% samples consisting of about 500,000 records obtained from the KDD Cup dataset is used in our study. We want to investigate on the parameters of MLP to implement weak classifiers. The main parameter which has sufficient impact on the complexity of MLP is the mean squared errors. As indicated in Figure 4, the detection rate of the weak classifier had been decreased by increasing the mean squared errors of MLP. Therefore, we select 0.2 as the mean squared errors to achieve 50% accuracy.

As indicated in figure5, increasing the mean squared errors of the weak classifier caused a decreasing manner in its training time. If we select 0.2 as the mean squared errors the run time of classification for 500000 records is approximately 650 seconds, i.e. the run time of classification for each record is about to 1.2 milliseconds.

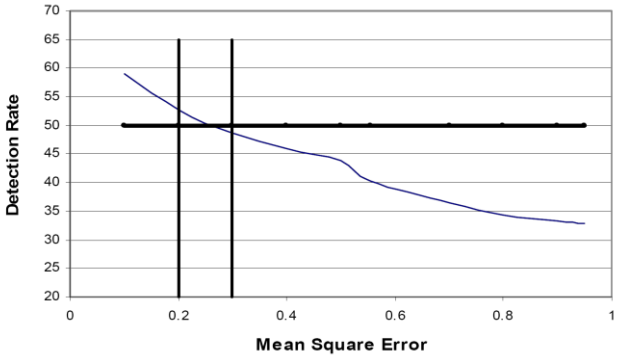

Figure 4: Detection rate versus mean squared errors.

[7]- The UCI Machine Learning Repository is a collection of databases, domain theories, and data generators that are used by the machine learning community for the empirical analysis of machine learning algorithms.

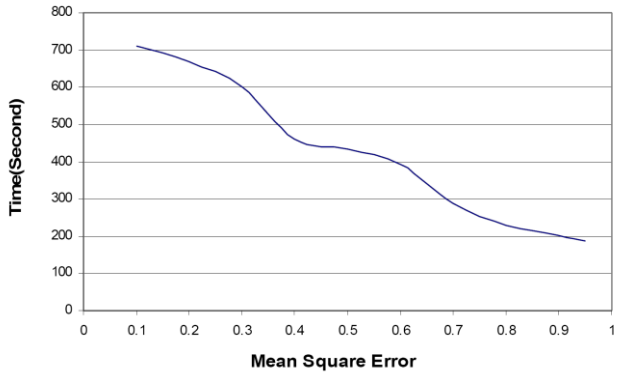

Figure 5: Time complexity versus mean squared errors.

Figure 6 shows the detection rate of the weak classifier when the number of epochs increases from 10 to 200. When the mean squared errors equals 0.2, the number of epochs must be set to 200 to obtain the 50% accuracy in classification.

Figure 7 shows the difference between real and desired output of the weak classifier when the number of epochs increases from 10 to 200. Figure 8 shows the detection rate of initial misuse detection model when the unknown threshold varies from 0.1 to 0.95. We select 0.8 to obtain accuracy near 50% for the weak learner.

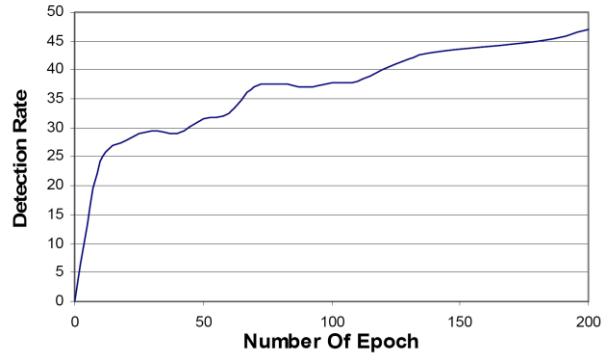

Figure 6: Detection rate versus the number of epoch.

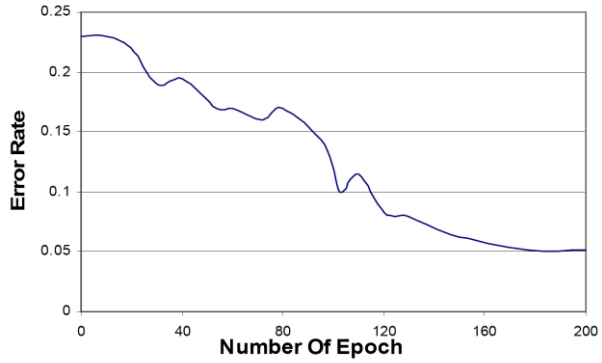

Figure 7: Error rate versus the number of epoch.

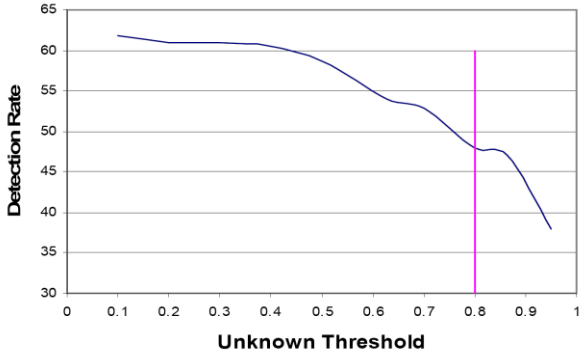

Figure 8: Detection rate versus unknown threshold.

B. Incremental Misuse Intrusion Detection System

We randomly select five training datasets and five testing datasets for evaluation of misuse intrusion detection. The distribution of training and testing dataset are shown in Table 4 and Table5, respectively.

TABLE 4
THE TRAINING DATA DISTRIBUTION FOR FIVE DATASET

dataset	DoS	U2R	R2L	PROBE	TOTAL
dataset1	30000	3	50	10000	40053
dataset2	40000	3	50	15000	55053
dataset3	50000	3	50	20000	70053
dataset4	70000	3	50	25000	95063
dataset5	100000	3	50	30000	130053

TABLE 5
THE TESTING DATA DISTRIBUTION FOR FIVE DATASET

dataset	DoS	U2R	R2L	PROBE	TOTAL
testset1	40000	7	150	20000	60157
testset2	60000	7	150	30000	90157
testset3	100000	7	150	40000	140157
testset4	180000	7	150	50000	230157
testset5	200000	7	150	70000	270157
Test	580000	35	750	210000	790157

Figures 9 and 10 show the detection rate of the intrusion detection system implemented with Learn++ and Adaboost.M1 when the number of weak hypotheses had been increased from 1 to 20. Because the new weak hypotheses is being trained based on the instances that are being misclassified by previous ensemble of weak hypotheses, so increasing the number of weak hypotheses caused the increase of the detection rate of the misuse intrusion detection system.

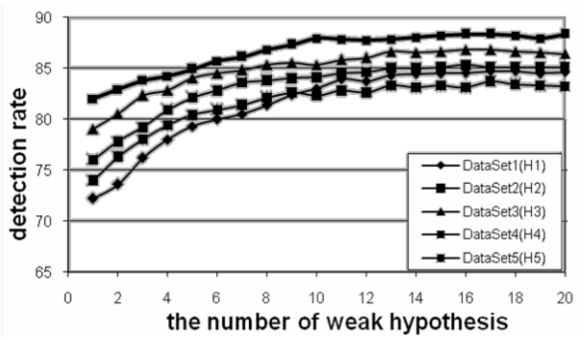

Figure 9: Detection rate versus the number of weak hypotheses for LEARN++ algorithm.

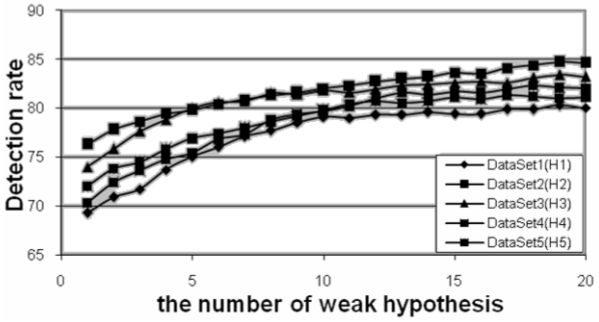

Figure 10: Detection rate versus the number of weak hypotheses for AdaBoost.M1 algorithm.

In order to investigate the incremental behavior of the misuse detection on different datasets, we divided each one of the five datasets into five subsections and construct five weak learners on each of them. Figure 11 and Figure 12 show that the detection rate of the intrusion detection is increased when increasing the number of the weak learner. Increasing the detection rate by adding weak learners shows the incremental behavior of the proposed system. In other words, the frameworks can learn new intrusions. The details of simulation are shown in Table 8 through Table 19 in appendix.

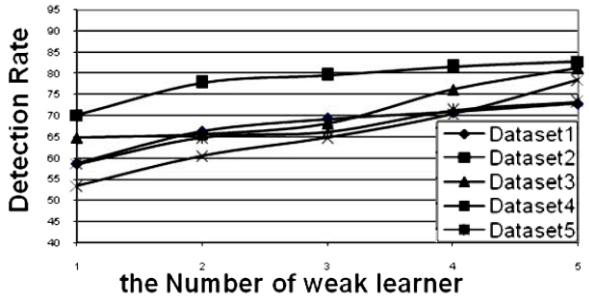

Figure 11: Detection rate of Adaboost.M1 algorithm versus the number of weak learner.

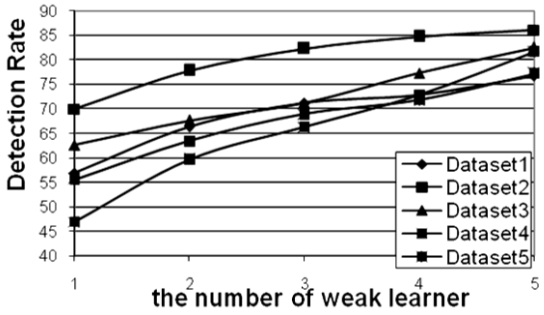

Figure 12: The classification performance of Learn++ on different dataset.

In order to show the sensitivity of the behavior of the intrusion detection system on the sequence of input data, we constructed three weak learners from the training dataset1 through dataset3 and investigated on the detection rate of both intrusion detection systems when the order of the weak learners which were applied to ensemble were changed. Figure 13 and Figure 14 show the results of our simulation. The details are shown in Table 20 through Table 31 in appendix.

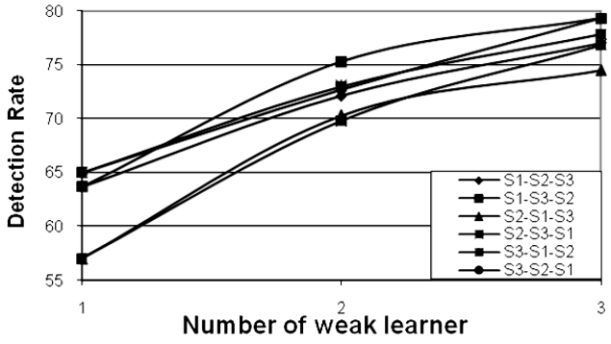

Figure 13: Detection rate of Learn++ versus the different sequence of weak learner.

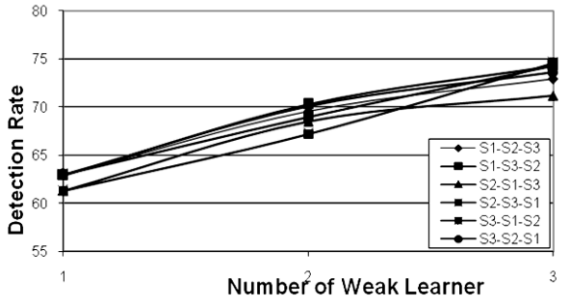

Figure 14: Detection rate of Adaboost.M1 versus the different sequence of weak learner.

C. Hybrid IDS with Manual Intrusion Classifier

The original training sample obtained from the KDD Cup 99 dataset was used in our study as the training set and the entire labeled test set was used for testing set. The

labeled test set has different distributions from the training set. We used the intrusions instances of training set to generate the model of incremental misuse detection system and used the normal instances of the training set to construct the profile of normal activities.

In order to generate the initial weak learner of our framework, the following scenario is done. To make the anomaly detection component, we elicit the normal instances from the training dataset for constructing the profile of normal activities. The remaining intrusions instances were used as the intrusion dataset to make the misuse detection component. We select the 10% of intrusion dataset consisting of about 400000 instances, which contains four class types of intrusions as initial dataset. This dataset will be used to make an initial model of misuse detection component.

After getting the initial weak learner of our framework, we pursue the following scenario for inserting additional models to the initial one. The 90% remaining intrusion dataset is applied to the initial model of misuse detection which has 21% detection rate on this dataset, and the others are being classified as suspicious activities. The unknown instances will be applied to the anomaly detection in order to detect unknown attacks, and then we manually determine the class types of instances that the anomaly detection component detects them as attacks. These new instances which are detected as attacks by the anomaly detection component are candidate for preparing a new intrusion dataset. In order to make the weak learner or model, we use a threshold equal to 100000 instances for constructing the next dataset which are randomly selected from the new intrusion dataset. The ensemble of the existing classifiers is used for the misuse detection component in the next iterations.

The above scenario will be done in several iterations for generating the new weak learner based on the new available intrusion dataset. The current model which generated from the new dataset and existing weak learners contribute to getting the final classification accuracy in the next iteration. At the end of the aforementioned scenario, 7 classifiers generated from the training dataset and the ensemble of them achieves 97% detection rate on the whole training dataset. The total number of instances used in our framework to generate 7 classifiers, are approximately 1000000 records, and the other instances are removed during the construction. Improving the detection rate from 21% to 97% on training dataset means that the existing model can learn new instances and achieve a higher classification accuracy.

After we get the new model of the misuse detection component based on the training dataset, we test the model on testing dataset for evaluating the effectiveness of our incremental intrusion detection system. The results of simulation show when a new weak learner is generated, the classification performance of ensemble approach on testing dataset increased. In other words, the framework can learn new information in the next iterations which becomes available in current iteration. Figure 15 shows increase of misuse detection rate on test dataset is 45.3 to 87.8 when the additional training sample is inserting for generating the weak learner. This increase demonstrates incremental learning capability of our scheme even when instances of new classes are introduced in subsequent training data. So, our hybrid intrusion detection system can learn new information incrementally.

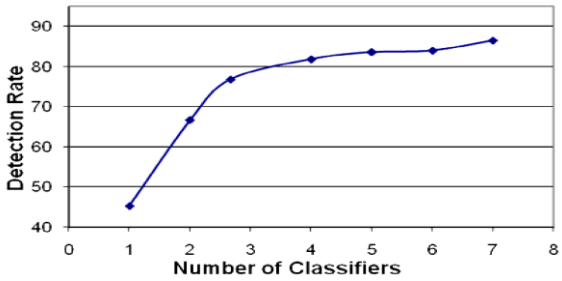

Figure 15: Classification performance versus the number of classifier.

The effectiveness of anomaly detection component on the performance classification of hybrid intrusion detection system is investigated in the following scenario. We remove the instances of data correctly classified by misuse detection component from the testing dataset and apply anomaly detection component on remaining instances of testing dataset. Many instances of remaining dataset detected as intrusions that misuse detection component could not clarify the classification output of them. As indicated in Figure 16 there are instances of data in remaining dataset that detected as intrusions by anomaly detection. These instances were not predicted by misuse detection component. This means that combining misuse detection and anomaly detection can detect more intrusions than each of them individually. These intrusions will be learned in the next iterations.

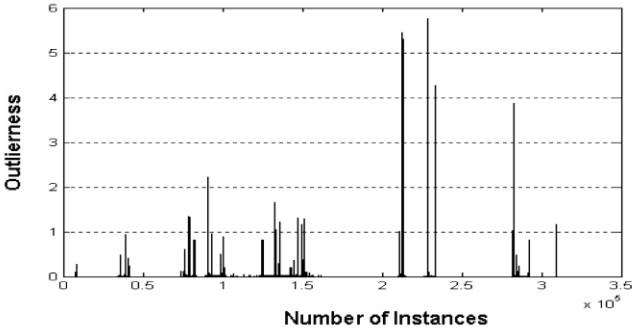

Figure 16: Outlier-ness versus the number of instances.

D. Hybrid IDS with Self-Labeled Intrusion Classifier

For simulations, the weak learner used to generate

individual hypotheses was a single hidden layer MLP with 41 hidden layer nodes and 5 nodes in output layer. The 4 nodes in output layer corresponded to the four class types of intrusions and the other corresponded to the new class type of intrusions which is detected by self-labeled intrusion classifier. The value of 0.2 was considered for mean squared errors of all MLPs to prevent over-fitting and to ensure sufficiently weak learning.

The 10% samples consisting of about 500,000 records obtained from the KDD Cup dataset was used in our study. The DoS attack type accounted for about 80% of the data, and consisted mostly of redundant Neptune and Smurf attacks. To reduce such a redundancy, the computational complexity of the problem and balancing between attack types, the dataset was reduced to 158491 records by down sampling over majority attacks and over sampling over minority attacks. These is done by removing duplicate Neptune and Smurf attack types from the DoS category and by increasing the U2R attacks up to 1000 instances. This process is adopted by other researchers such as [27]. The U2R type was almost negligible. We used unsupervised clustering algorithms for anomaly detection. The labels were not used during the clustering process, but were used for evaluating the detection performance of the algorithms.

We divided the 10% of training dataset in two datasets: intrusion dataset and normal dataset. Three models were constructed from these datasets.

The initial model of misuse detection constructed from intrusion dataset

The initial model of anomaly detection constructed from normal dataset.

The initial model of the *self-labeled intrusion classifier* constructed from the intrusion dataset.

In order to obtain the model of the self-labeled intrusion classifier, we used the intrusion dataset. At first we ignore the label of instances and then applying *on-line k-mean* on them. Then based on the purity of cluster for the intrusion types, we assigned clusters to classes types of intrusions. Table 6 shows the detection rate of the self-labeled intrusion classifier before and after down and over sampling. The results show that after down and over sampling the detection rate of Probe and DoS attacks were decreased and the detection rate of U2R and R2L attacks were increased.

TABLE 6
DETECTION RATE OF COMPONENT USED TO DETERMINE THE CLASS TYPES OF INTRUSIONS

Sampling Algorithm	Number of Cluster	False Positive	Detection Rate			
			Probe	DoS	U2R	R2L
without over and down	70	0.364	99.7	100	53.6	87.3
with over and down	70	0.231	98.8	97.2	83.7	92.9

7. COMPLEXITY ANALYSIS AND COMPARISON

After analysing the LEARN++[21], we calculate it in the training phase of our framework. The computational complexity of the initial model in the on-line phase is $O(nT_k\alpha)$, where n is the number of instances, T_k is the number of weak hypotheses that must be generated, α is the complexity of weak classifier which in our framework is simple multi layer perceptron. For testing phase, computational complexity of our framework is $O(n\alpha')$, where n is the number of test instances, and α' is the complexity of weak hypotheses in testing phase. The computational complexity of the training phase depends on the distribution of dataset, and in the worst case it is $O(n^2M^2)$, which is higher than Learn++. M is the number of decision stumps. In other word, Learn++ generally possesses lower computational complexity than strong artificial neural network, especially in training phase.

Clustering algorithms can be divided in to two categories [40]: similarity based and centroid based. Similarity algorithms have the complexity of $O(N^2)$, where N is the number of instances. Contrast centroid-based algorithms have a complexity of $O(NKM)$, where K is the number of clusters, M is the number of batch iteration, and N is the number of instances. The on-line k-mean algorithm is a centroid-based algorithm which can be a desirable choice for on-line learning, because it has high clustering quality, relatively low complexity and fast convergence.

Table 7 shows the comparison to the other algorithms which were be implemented on KDD Cup Dataset 99. It shows that the proposed framework with over and down sampling has a better detection rate in the average. Our proposed framework has high clustering quality, relatively low computational complexity and fast convergence.

TABLE 7
COMPARISION OF THE PROPOSED FRAMEWORK WITH OTHER ALGORITHMS

Algorithmes	Detection Rate				
	Probe	DoS	U2R	R2L	Average
EFuNN [28]	99.88	98.99	65.00	97.26	90.28
Ensemble (DT, SVM) [19]	98.57	99.92	48.00	37.80	71.07
Ensemble (DT, SVM, EN (DT, SVM)) [19]	100	99.82	68.00	97.16	91.25
Hybrid multi-level tree classifier [33]	99.71	99.19	66.67	89.50	88.77
Our framework without over and down sampling	99.7	100	53.60	87.31	85.15
Our framework with over and down sampling	98.8	97.2	83.70	92.90	93.15

8. CONCLUSION

In the case that the intrusion detection instances are updated continually and infinitely, the static model learning on the initial training dataset unable to update the profile of model dynamically. For improving adaptively to network behavior, we presented an incremental hybrid intrusion detection model based on ensemble of weak classifiers. The detecting model can incorporate new instances continually, and therefore enhance generalization performance of the detecting model.

We used boosting ensemble of weak classifiers to focus on the detecting and learning of new intrusions which belong to the hard-to-learn instances. Proposed intrusion detection systems using ensemble of weak classifiers generally possesses lower computational complexity than other frameworks which used strong classifiers.

The research of this paper will have an important significance for building an efficient and applicable intrusion detection system. We used unsupervised clustering to implement the self-labeled intrusion classifier. In future we want to use supervised clustering to implement it; with this manner self-labeled intrusion component will detect new intrusions.

9. REFERENCES

[1] D. Anderson, T. Frivold, and A. Valdes, "Next-Generation Intrusion Detection Expert System", (NIDES)-A Summary, Technical Report SRICLS-95-07, SRI, May 1995.

[2] D. Barbarra, J. Couto, S. Jajodia, L. Popyack, and N. Wu, "ADAM: Detecting Intrusion by Data Mining", Proceedings of the 2001 IEEE, Workshop on Information Assurance and Security T1A3 1100 United States Military Academy, West Point, NY, June 2001.

[3] C. Amza, C.Leordeanu, V. Cristea, "Hybrid network Intrusion Detection ", IEEE International Conference on Intelligent Computer Communication and Processing (ICCP), 2011, Page(s): 503 – 510..

[4] DARPA Intrusion detection evaluation: http://www.ll.mit.edu/SSt/ideval/result/result_index.html.

[5] O. Depren, M. Topallar, E. Anarim, and M. K. Ciliz, "An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks", Expert Systems with Applications Volume 29, Issue 4, Pages 713-722, , November 2005.

[6] E. Eleazar, "Anomaly Detection over Noisy Data using Learned Probability Distributions", ICML00, Palo Alto, CA: July, 2000.

[7] Y. Freund and R. Schapire, "A decision theoretic generalization of on-line learning and an application to boosting", Comput. Syst. Sci., vol. 57, no. 1, pp. 119–139, 1997.

[8] G. Giacinto, F. Roli, and L. Didaci, "Fusion of multiple classifiers for intrusion detection in computer networks", Pattern Recognition Letters, 24(12), pp. 1795-1803, 2003.

[9] R. Heady, G. Luger, A. Maccabe, and M. Servilla. "The architecture of a network level intrusion detection system", Technical Report CS90-20, Department of Computer Science, University of New Mexico, August 1990.

[10] W. Hu and W. Hu, "Network-based Intrusion Detection Using Adaboost Algorithm", Proceedings of the 2005 IEEE/WIC/ACM International conference on Web Intelligence(WI'05), 0-7695-2415-X/05, 2005.

[11] K. Hwang, M. Cai, Y. Chen, and M. Qin, "Hybrid Intrusion Detection with Weighted Signature Generation over Anomalous Internet Episodes", IEEE Transaction on Dependable and Secure Computing , Vol. 4, No. 1, pp. 41-55, January-March 2007.

[12] KDD Cup 1999 Intrusion detection dataset, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[13] K. Leung and C. Leckie, "Unsupervised Anomaly Detection in Network Intrusion Detection Using Clusters", Australasian Computer Science Conference, Newcastle, NSW, Australia, 2005.

[14] P. Lichodzijewski, A.N. Zincir-Heywood, and M. I. Heywood, "Host-based intrusion detection using self-organizing maps," Proceedings of the 2002 IEEE World Congress on Computational Intelligence, 2002.

[15] U. Lindqvist and P.A. Porras, "Detecting computer and network misuse through the production-based expert system toolset (PBEST)", Proceedings of the 1999 IEEE symposium on security and privacy, pp. 146-161, IEEE Computer Socitey, Los Alamitos, CA., 1999.

[16] N. Littlestone and M.Warmuth, "Weighted majority algorithm", Inform. Comput. vol. 108, pp. 212–261, 1994.

[17] M. Locasto, K. Wang, A. Keromytis, and S. Stolfo. Flips: Hybrid adaptive intrusion prevention. In Proceedings of the 8th International Symposium on Recent Advances in Intrusion Detection (RAID), September 2005.

[18] Mounji, B.L. Charlier, D. Zampuniéris, and N. Habra, "Distributed audit trail analysis:, Proceedings of the ISOC'95 symposium on network and distributed system security, pp. 102-112, IEEE Computer Society, Los Alamitos, CA., 1995.

[19] S. Peddabachigaria, A. Abrahamb, I. Grosanc, and J. Thomas, "Modeling intrusion detection system using hybrid intelligent systems", Published by Elsevier Ltd, 2005.

[20] S. Peddabachigaria, A. H. Sung, and A. Abraham, "Intrusion detection using an ensemble of intelligent paradigms", Published by Elsevier Ltd, 2004.

[21] R. Polikar, L. Udpa, and V. Honavar, "Learn++: An incremental learning algorithm for supervised neural networks", IEEE Transactions on System, Man and Cybernetics (C), Special Issue on Knowledge Management, vol. 31, no. 4, pp. 497-508, 2001.

[22] P. Porras and G. P. Neumann, "EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances", In Proceedings of 20th National Information Systems Security Conference, 1997.

[23] S.T. Powers and J. He, "A hybrid artificial immune system and Self Organizing Map for network intrusion detection", Information Sciences 178, pp. 3024–3042, 2008.

[24] R. Rangadurai Karthick, V.P.Hattiwale, B. Ravindran, "Adaptive network intrusion detection system using a hybrid approach", Fourth International Conference on Communication Systems and Networks (COMSNETS), 2012, Page(s): 1 – 7.

[25] Rasoulifard and A. Ghaemi Bafghi, "Incremental Intrusion Detection Using Learn++ algorithm", 3rd conference on Information and Knowledge Technology, Ferdowsi University of Mashhad, Faculty of Engineering, IKT2007, Nov. 27-29 2007.

[26] Rasoulifard, A. Ghaemi Bafghi, and M. kahani, "Incremental Hybrid Intrusion Detection Using Ensemble of Weak Classifiers", 13th Int'l CSI Computer Conference (CSICC'08), March 9-11, 2008.

[27] M. Sabhnani and G. Serpen, "Application of Machine Learning Algorithms to KDD Intrusion Detection Dataset within Misuse Detection Context", EECS Dept, University of Toledo, Toledo, Ohio 43606 USA.

[28] K. Shah, N. Dave, S. Chavan, S. Mukherjee, A. Abraham, and S. Sanyal, "Adaptive Neuro-Fuzzy Intrusion Detection System", IEEE International Conference on ITCC'04, Vol. 1, pp. 70-74, 2004.

[29] K, Selvamani; S, Anbuchelian; S, Kanimozhi; R, Elakkiya; S, Bose; A, Kannan, "A hybrid framework of intrusion detection system for resource consumption based attacks in wireless ad-hoc

networks", International Conference on Systems and Informatics (ICSAI), 2012, Page(s): 8 – 12..
[30] T. Shon and J. Moon, "A hybrid machine learning approach to network anomaly detection", Information Sciences 177, pp.3799–3821, 2007.
[31] E. Tombini, H. Debar, L. Mé, and M. Ducassé, "A Serial Combination of Anomaly and Misuse IDSes Applied to HTTP Traffic", In proceedings of the Annual Computer Security Applications Conference (ACSAC). December 2004.
[32] K. Wang and S. J. Stolfo. "Anomalous Payload-based Network Intrusion Detection", In Proceedings of the 7th International Symposium on Recent Advances in Intrusion Detection (RAID), pages 203-222, September 2004.
[33] Xiang and S.M. Lim, "Design of Multiple-Level Hybrid Classifier for Intrusion Detection System", Proceeding of Machine Learning for Signal Processing, 2005 IEEE Workshop on Volume , Issue , 28-28 ,PP 117 – 122, Sept. 2005.
[34] L. Xu, A. Krzyzak, and Y.Ching, "Methods of Combining Multiple Classifier and Their Application to Handwriting Recognition", IEEE TRANSACTION ON SYSTEMS, MAN AND CYBERNETICS, VOL. 22, NO. 3, MAY/JUNE 1992.
[35] W. Yang, X.C. Yun, and L.J. Zhang, "Using Incremental Learning Method For Adaptive Network Intrusion Detection", Proceedings of the Fourth International Conference on Machine Learning and Cybernetics, Guangzhou, 18-21 August 2005.
[36] Z. Yu and J.P. Tsai, "A Multi-Class SLIPPER System for Intrusion Detection," compsac, pp. 212-217, 28th Annual International Computer Software and Applications Conference (COMPSAC'04), 2004.
[37] Z. Yu and J.P. Tsai, "An efficient intrusion detection system using a boosting-based learning algorithm," International Journal of Computer Applications in Technology, Vol. 27, No.4 pp. 223 – 231, 2006.
[38] J. Zhang and M. Zulkernine, "Anomaly based network intrusion detection with unsupervised outlier detection", The 2006 IEEE International Conference on Communications, Istanbul, Turkey, June 2006.
[39] J. Zhang and M. Zulkernine, "A Hybrid Network Intrusion Detection Technique Using Random Forests", Proc. of the International Conference on Availability, Reliability and Security (AReS), IEEE CS Press, pp. 262-269, Vienna, Austria, April 2006.
[40] S. Zhong, T. Khoshgoftaar, and N. Seliya, "Clustering-Based Network Intrusion Detection", International Journal of Reliability, Quality and Safety Engineering.

10. APPENDIX

TABLE 8
TRAINING AND GENERALIZATION PERFORMANCE OF LEARN++ ALGORITHMS ON DATASET1

Dataset	H1	$\sum(H1,H2)$	$\sum(H1,...,H3)$	$\sum(H1,...,H4)$	$\sum(H1,...,H5)$
S1	96.4%	90.8%	88.4%	83.1%	80.7%
S2	---	83.6%	75.9%	71.6%	68.5%
S3	---	---	94.8%	88.5%	84.8%
S4	---	---	---	87.4%	77.1%
S5	---	---	---	---	83.65
Test	56.9%	66.4%	71.2%	72.8%	76.8%

TABLE 9
TRAINING AND GENERALIZATION PERFORMANCE OF LEARN++ ALGORITHMS ON DATASET2

Dataset	H1	$\sum(H1,H2)$	$\sum(H1,...,H3)$	$\sum(H1,...,H4)$	$\sum(H1,...,H5)$
S1	98.8%	92.7%	89.3%	86.3%	85.2%
S2	---	91.3%	87%	83%	81.6%
S3	---	---	91%	89.7%	87.2%
S4	---	---	---	93.4%	89.4%
S5	---	---	---	---	95.6%
Test	70%	77.8%	82.3%	84.7%	86.0%

TABLE 10
TRAINING AND GENERALIZATION PERFORMANCE OF LEARN++ ALGORITHMS ON DATASET3

Dataset	H1	$\sum(H1,H2)$	$\sum(H1,...,H3)$	$\sum(H1,...,H4)$	$\sum(H1,...,H5)$
S1	90.0%	84.3%	79.3%	78.6%	77.6%
S2	---	96.1%	92.4%	84.3%	82.1%
S3	---	---	91.3%	89.7%	88.3%
S4	---	---	---	90.1%	88.7%
S5	---	---	---	---	89.0%
Test	62.6%	67.6%	71.2%	77.3%	82.5%

TABLE 11
TRAINING AND GENERALIZATION PERFORMANCE OF LEARN++ ALGORITHMS ON DATASET4

Dataset	H1	$\sum(H1,H2)$	$\sum(H1,...,H3)$	$\sum(H1,...,H4)$	$\sum(H1,...,H5)$
S1	97.2%	93.6%	79.3%	85.2%	81.3%
S2	---	92.7%	92.4%	82.1%	79.4%
S3	---	---	91.3%	88.3%	86.3%
S4	---	---	---	96.7%	89.6%
S5	---	---	---	---	92.1%
Test	47%	59.7%	66.3%	72.8%	81.7%

TABLE 12
TRAINING AND GENERALIZATION PERFORMANCE OF LEARN++ ALGORITHMS ON DATASET5

Dataset	H1	$\sum(H1,H2)$	$\sum(H1,...,H3)$	$\sum(H1,...,H4)$	$\sum(H1,...,H5)$
S1	88.0%	83.2%	78.3%	74.2%	72.6%
S2	---	92.3%	87.2%	82.7%	80.3%
S3	---	---	88.6%	84.3%	79.2%
S4	---	---	---	93.0%	87.4%
S5	---	---	---	---	88.8%
Test	55.6%	63.4%	68.9%	71.8%	77.2%

TABLE 13
THE PERFORMANCE CLASSIFICATION OF LEARN++ ON DIFFERENT DATASET

Dataset	Min	Max	Avg
Test	76.8%	88.0%	81.2%

TABLE 14
TRAINING AND GENERALIZATION PERFORMANCE OF ADABOOST.M1 ALGORITHMS ON DATASET1

Dataset	H1	$\sum(H1,H2)$	$\sum(H1,...,H3)$	$\sum(H1,...,H4)$	$\sum(H1,...,H5)$
S1	93.7%	91.2%	88.4%	83.1%	82.7%
S2	---	89%	75.9%	71.6%	66.8%
S3	---	---	94.8%	88.5%	81.9%
S4	---	---	---	87.4%	78.1%
S5	---	---	---	---	86.5%
Test	58.7%	66.2%	69.2%	70.8%	72.8%

TABLE 15
TRAINING AND GENERALIZATION PERFORMANCE OF ADABOOST.M1 ALGORITHMS ON DATASET2

Dataset	H1	$\sum(H1,H2)$	$\sum(H1,...,H3)$	$\sum(H1,...,H4)$	$\sum(H1,...,H5)$
S1	97.3%	91.3%	88.6%	83.3%	83.9%
S2	---	92.6%	86.8%	81.7%	80..3%
S3	---	---	90.5%	88.6%	86..2%
S4	---	---	---	91.9%	87.3%
S5	---	---	---	---	93.8%
Test	70%	77.8%	79.5%	81.6%	82..7%

TABLE 16
TRAINING AND GENERALIZATION PERFORMANCE OF ADABOOST.M1 ALGORITHMS ON DATASET3

Dataset	H1	$\sum(H1,H2)$	$\sum(H1,...,H3)$	$\sum(H1,...,H4)$	$\sum(H1,...,H5)$
S1	96.%	86.3%	80.3%	78.6%	77.6%
S2	---	94.1%	89.4%	84.3%	82.1%
S3	---	---	90.5%	89.7%	88.3%
S4	---	---	---	90.1%	88.7%
S5	---	---	---	---	89.0%
Test	64.7%	65.6%	68.2%	76.1%	81.2%

TABLE 17
TRAINING AND GENERALIZATION PERFORMANCE OF ADABOOST.M1 ALGORITHMS ON DATASET4

Dataset	H1	$\sum(H1,H2)$	$\sum(H1,...,H3)$	$\sum(H1,...,H4)$	$\sum(H1,...,H5)$
S1	96.3%	88.8%	83.3%	83.9%	80.6%
S2	---	93.6%	89.3%	83.1%	78.5%
S3	---	---	90.8%	84.3%	83.3%
S4	---	---	---	92.7%	82.6%
S5	---	---	---	---	90.1%
Test	53.4%	60.5%	64.7%	70.3%	78.4%

TABLE 18
TRAINING AND GENERALIZATION PERFORMANCE OF ADABOOST.M1 ALGORITHMS ON DATASET5

Dataset	H1	$\sum(H1,H2)$	$\sum(H1,...,H3)$	$\sum(H1,...,H4)$	$\sum(H1,...,H5)$
S1	82.3%	81.2%	77.8%	74.2%	72.6%
S2	---	89.8%	86.3%	82.7%	80.3%
S3	---	---	88.6%	84.3%	79.2%
S4	---	---	---	93.0%	87.4%
S5	---	---	---	---	88.8%
Test	58.6%	64.8%	66.1%	71.2%	73.2%

TABLE 19
THE PERFORMANCE CLASSIFICATION OF ADABOOST.M1 ON DIFFERENT DATASET

Dataset	Min	Max	Avg
Test	72.8%	82.7%	77.66%

TABLE 20
TRAINING AND GENERALIZATION PERFORMANCE OF LEARN++ ALGORITHMS ON SEQUENCE S1, S2, S3

Dataset	H1	$\sum(H1,H2)$	$\sum(H1,...,H3)$
S1	97.6%	77.6%	75.6%
S2	---	87.5%	83.7%
S3	---	---	94.0%
Test	63.7%	72.1%	77.0%

TABLE 21
TRAINING AND GENERALIZATION PERFORMANCE OF LEARN++ ALGORITHMS ON SEQUENCE S1, S3, S2

Dataset	H1	$\sum(H1,H2)$	$\sum(H1,...,H3)$
S1	97.6%	82.4%	78.2%
S3	---	81.9%	84.4%
S2	---	---	91.1%
Test	63.7%	75.3%	79.3%

TABLE 22
TRAINING AND GENERALIZATION PERFORMANCE OF LEARN++ ALGORITHMS ON SEQUENCE S2, S1, S3

Dataset	H1	$\sum(H1,H2)$	$\sum(H1,...,H3)$
S2	94.2%	76.8%	76.9%
S1	---	83%	81.2%
S3	---	---	90.7%
Test	57.0%	70.3%	74.5%

TABLE 23
TRAINING AND GENERALIZATION PERFORMANCE OF LEARN++ ALGORITHMS ON SEQUENCE S2, S3, S1

Dataset	H1	$\sum(H1,H2)$	$\sum(H1,...,H3)$
S2	94.2%	72.7%	77.2%
S3	---	81.4%	80.4%
S1	---	---	91.2%
Test	57.0%	69.8%	76.8%

TABLE 24
TRAINING AND GENERALIZATION PERFORMANCE OF LEARN++ ALGORITHMS ON SEQUENCE S3, S1, S2

Datasets	H1	$\sum(H1,H2)$	$\sum(H1,...,H3)$
S3	92.7%	82.4%	78.0%
S1	---	86.0%	83.3%
S2	---	---	92.1%
Test	65.0%	73.0%	77.8%

TABLE 25
TRAINING AND GENERALIZATION PERFORMANCE OF LEARN++ ALGORITHMS ON SEQUENCE S3, S3, S1

Dataset	H1	$\sum(H1,H2)$	$\sum(H1,...,H3)$
S3	92.7%	81.7%	79.5%
S2	---	87.0%	81.0%
S1	---	---	91.3%
Test	65.0%	72.7%	79.3%

TABLE 26
TRAINING AND GENERALIZATION PERFORMANCE OF ADABOOST ALGORITHMS ON SEQUENCE S1, S2, S3

Dataset	H1	$\sum(H1,H2)$	$\sum(H1,...,H3)$
S1	96.8%	74.6%	75.6%
S2	---	84.%	80.6%
S3	---	---	91.4%
Test	57.9%	69.6%	72.9%

TABLE 27
TRAINING AND GENERALIZATION PERFORMANCE OF ADABOOST ALGORITHMS ON SEQUENCE S1, S3, S2

Dataset	H1	$\sum(H1,H2)$	$\sum(H1,...,H3)$
S1	96.8	79.2%	77.0%
S3	---	81.9%	76.6%
S2	---	---	88.7%
Test	57.9%	70.3%	74.3%

TABLE 28
TRAINING AND GENERALIZATION PERFORMANCE OF ADABOOST ALGORITHMS ON SEQUENCE S2, S1, S3

Dataset	H1	$\sum(H1,H2)$	$\sum(H1,...,H3)$
S2	96.8%	83.8%	72.9%
S1	---	86.2%	79.2%
S3	---	---	91.3%
Test	61.3%	68.5%	71.2%

TABLE 29
TRAINING AND GENERALIZATION PERFORMANCE OF ADABOOST ALGORITHMS ON SEQUENCE S2, S3, S1

Datasets	h1	$\sum(H1,H2)$	$\sum(H1,...,H3)$
S2	96.8%	76.8%	75.8%
S3	---	79.4%	78.8%
S1	---	---	90.8%
Test	61.3%	67.2%	74.6%

TABLE 30
TRAINING AND GENERALIZATION PERFORMANCE OF ADABOOST ALGORITHMS ON SEQUENCE S3, S1, S2

Dataset	H1	$\sum(H1,H2)$	$\sum(H1,...,H3)$
S3	93.4%	79.5%	76.4%
S1	---	85.1%	81.2%
S2	---	---	90.3%
Test	63.0%	69.0%	74.3%

TABLE 31
TRAINING AND GENERALIZATION PERFORMANCE OF ADABOOST ALGORITHMS ON SEQUENCE S3, S3, S1

Dataset	H1	$\sum(H1,H2)$	$\sum(H1,...,H3)$
S3	93.4%	81.7%	79.5%
S2	---	87.0%	81.0%
S1	---	---	91.3%
Test	63.0%	70.1%	73.6%