A Novel Statistical Technique for Intrusion Detection Systems

Enamul Kabir, Jiankun Hu, and Hua Wang, and Guangping Zhuo[*][†][‡][§]

Abstract

This paper proposes a novel approach for intrusion detection system based on sampling with Least Square Support Vector Machine (LS-SVM). Decision making is performed in two stages. In the first stage, the whole dataset is divided into some predetermined arbitrary subgroups. The proposed algorithm selects representative samples from these subgroups such that the samples reflect the entire dataset. An optimum allocation scheme is developed based on the variability of the observations within the subgroups. In the second stage, least square support vector machine (LS-SVM) is applied to the extracted samples to detect intrusions. We call the proposed algorithm as optimum allocation-based least square support vector machine (OA-LS-SVM) for IDS. To demonstrate the effectiveness of the proposed method, the experiments are carried out on KDD 99 database which is considered a de facto benchmark for evaluating the performance of intrusions detection algorithm. All binary-classes and multiclass are tested and our proposed approach obtains a realistic performance in terms of accuracy and efficiency. Finally a way out is also shown the usability of the proposed algorithm for incremental datasets.

keywords: Sampling, Intrusion Detection System (IDS), Network Security, Least Square Support Vector Machine (LS-SVM).

1 Introduction

In recent years, there has been an increasing awareness of the risk associated with network attacks as information systems are now more open to the Internet than ever before. Intrusion detection system (IDS) is a program that tries to find indications that the computer has been compromised. An IDS attempts to detect an intruder breaking into computer system or legitimate user misuses system resources. Intrusion detection is an important issue and has captured the attention of network administrators and security professionals.

Intrusion detection is the art of detecting unauthorized, inappropriate, or anomalous activity on computer systems. Intrusion detection systems are classified as network based,

[*]Enamul Kabir is with the School of Agricultural, Computational and Environmental Sciences, University of Southern Queensland, Toowoomba, QLD 4350, Australia. E-mail: Enamul.Kabir@usq.edu.au

[†]Jiankun Hu is with the School of Engineering and Information Technology, University of New South Wales at the Australian Defence Force Academy (UNSW@ADFA), Canberra, ACT 2600,Australia. E-mail: J.Hu@adfa.edu.au

[‡]Hua Wang is with Centre for Applied Informatics, Victoria University, Melbourne, VIC 8001, Australia. E-mail: Hua.Wang@vu.edu.au

[§]Guangping Zhuo is with Department of Computer Science, Taiyuan Normal University, China. E-mail: zhuoguangping@163.com host based, or application based depending on their mode of deployment and data used for analysis [1, 35]. In addition, intrusion detection systems can also be classified as signature based or anomaly based depending upon the attack detection method. The signature-based systems are trained by extracting specific patterns (or signatures) from previously known attacks while the anomaly-based systems learn from the normal data collected when there is no anomalous activity [1, 30, 31, 32, 43]. The main purpose of an IDS is to detect as many attacks as possible with minimum number of false alarms, i.e., the system must be accurate in detecting attacks. However, an accurate system that cannot handle large amount of network traffic and is slow in decision making will not fulfill the purpose of an intrusion detection system [18]. Hence it is necessary to develop a system that detects most of the attacks, gives very few false alarms, copes with large amount of data, and is fast enough to make real-time decisions.

Although the IDS has led to a number of valuable network security techniques [3, 4, 14, 15, 16, 17, 18, 33, 40, 41, 42], the existing solutions are limited only to static data release. That is, in such solutions it is assumed that the entire dataset is available at the time of release. This assumption implies a significant shortcoming, as data today are continuously collected (thus continuously grow) and there is a strong demand for up-to-date data at all times. One possible approach is to use the intrusion detection techniques for the entire dataset whenever the dataset is augmented with new records. In this way, researchers are always provided with up-to-date information. Although this can be accomplished using existing techniques, there are two significant drawbacks. First, it requires redundant computation, as the entire dataset has to be analysed even if only a few records are newly inserted. Sometimes intrusion detection techniques might not work properly due to continuously growing large dataset. Secondly, huge space will be required to store all the previous datasets that may be sometimes impossible. So it is necessary to develop an IDS system that can be used for static as well as for incremental datasets.

In the past few years, many researches have tried to apply different techniques for detecting intrusions. Among them, the framework of support vector machines (SVM) is becoming extremely popular in the field of statistical pattern classification. Least square support vector machines (LS-SVM) are the modified version of support vector machines. LS-SVM has been used different purposes such as for adaptive communication channel equalization [19], to study the nonlinear time series prediction [20], on Morlet Wavelet kernel function [21], for facial gender classification [22] and for measurement of soluble solids content of rice vinegars [24]. Although LS-SVM is significant, it has not yet being used for detecting intrusions. This paper proposes a LS-SVM technique in order to detect intrusions for IDS both in static and incremental datasets.

From the pattern recognition point of view the key problem is to represent the large amount of dataset for further analysis, such as classification. It is important to extract useful features from the large datset and then use the extracted features for classification. In the literature, numerous intrusion detection techniques are often employed for the feature extraction and the classification stage. The main drawback of these methods is that they do not work well when the data size is very large. They also require lengthy training time.

Table 1: Required sample size

Population	Sample	
	$99-100\%$ confidence interval	
	95% confidence level	99% confidence level
100	99	99
1K	906	943
10K	4899	6247
100K	8763	14267
1M	9513	16369
10M	9595	16613
100M	9603	16638
1B	9604	16641

Observing this challenge, this paper proposes an optimum allocation-based least square support vector machine (OA-LS-SVM) for IDS. The proposed approach uses the idea of sampling as representative samples can describe the whole population. For a given population, if the sample size is adequately taken then it can tell the characteristics of the population. Now a natural question arises, how many samples are required to describe the whole population? The process of determining sample size used in this paper are described in subsection 3.1. Table 1 shows the required sample size from a specified population under 99-100% confidence interval and for both 95% and 99% confidence levels using equation 3.2 in subsection 3.1.

As shown in Table 1, the increment of the sample size is not the same as the population size. If the population size is 100 Millions, we need a sample of size 9603 under 99-100% confidence interval and 95% confidence level whereas we need one more sample if the population size is 1 Billion. Thus it is a natural expectation that the sampling process can be used for intrusion detection for large datasets. This expectation is achieved in this paper for detecting intrusions. The OA-LS-SVM algorithm proposed in this paper consists of the following steps:

1. Combine the training and testing dataset and determine the required size of the sample by using the equation 3.2 in subsection 3.1 under desired confidence interval and confidence level.

2. Determine the size of training and testing using optimum allocation (OA) scheme as discussed in Section 3.

3. Divide the training and testing dataset into some predetermined subgroups of arbitrary instances. Using OA scheme in Section 3, select instances from each group of training and testing dataset such that the sum of these instances is equal to the desired sizes of respective training and testing as in Step 2.

4. The selected instances in step 3 will be used as an input set in LS-SVM to detect different intrusions.

The reminder of this paper is organized as follows. Section 2 reviews related work on intrusion detection system. We present a description of the proposed methodology in details in Section 3 both for static and incremental data. Section 4 shows experimental results of the proposed method. Finally, concluding remarks are included in Section 5.

2 Related Work

This work is related to several topics in the area of network security in information detection systems. Considering the risk associated with network attacks, a number of methods and frameworks have been proposed and many systems have been developed to detect intrusions. This section briefly discusses these techniques and framework.

Very Recently Gupta et al. [18] proposed an intrusion detection system using conditional random fields (CRF) and Layered approach. They considered the attack categories as layers and different features were selected for each layer. The dataset was divided into five attack categories for training and testing purposes of each layer. The test data passed through the cascaded layers to determine the category a record belonged to. This approach is however effective only for the selected features but not so convincing considering all features. On the other hand, the results from automatic feature selection is not so promising as manual selection. Thus the feature selection is a critical issue and so the practical implementation of this approach is limited. Data mining approaches for detecting intrusions was introduced by Lee et al. [4, 5, 7]. It include association rules [12] and frequent episodes, which are based on building classifiers by discovering relevant patterns of program and user behaviour. These methods can deal with symbolic data, and the features can be defined in the form packet and connection details. However mining of features is limited to entry level of the packet and requires the number of records to be large and sparsely populated; otherwise, they tend to produce a large number of rules that increase the complexity of the system [11].

$k-$ means [9, 34, 39] and the fuzzy c-means [10] have been applied extensively for intrusion detection. The main drawback of these clustering based techniques are that they are based on calculating numeric distance between the observations, and hence the observations must be numeric and thus observations with symbolic features cannot be easily used. On the other hand, the clustering methods consider the features independently and are unable to capture the relationship between different features of single record, which further degrades attack detection accuracy. Amor et al. [6] used Naïve Bayes classifiers for intrusion detection. The authors make strict independence assumption between the features in an observation resulting in lower attack detection accuracy when the features are correlated, which is often the case for intrusion detection. Bayesian network [13] has also been used for intrusion detection. This method tends to be attack specific and build a decision network based on special characteristics of individual attacks. Thus the size of a Bayesian network increases rapidly as the number of features and the type of attacks modeled by a Bayesian network increases.

Hidden Markov models (HMMs) have also been applied for intrusion detection [23, 27, 29]. However, modeling the system calls alone may not always provide accurate classification as in such cases various connection level features are ignored. Further, HMMs are generative systems and fail to model long-range dependencies between the observations [28]. Debar et al. [36] and Zhang et al [38] discussed the use of artificial neural networks for network intrusion detection. The main drawback of these methods is that they require large amount of data for training and it is hard to select best possible architecture for a neural network. The idea of decision trees have also used for intrusion detection [6]. It generally has high speed of operation and high attack detection accuracy. Kim et al. [17] used the support vector machine for intrusion detection. Support vector machines (SVMs) map real valued input feature vector to a higher dimensional feature space through nonlinear mapping and can provide real-time detection capability, deal with large dimensionality of data, and can be used for binary-class as well as multiclass classification.

The work presented in this paper uses the idea of least square support vector machine (LS-SVM) which is a modified version of SVM. For large datasets, it is necessary to reduce the dimension of the dataset and fed them to classifiers to detect intrusion. We first determine the required size to describe the characteristics of the whole dataset. Then we divide the whole dataset into some predetermined subgroups and select sample from these clusters using the derived optimum allocation scheme. We finally use these samples as an input set of LS-SVM to detect different attacks in IDS. We compare our approaches with the most recent methods of intrusion detection in the literature. The difference between the pro-

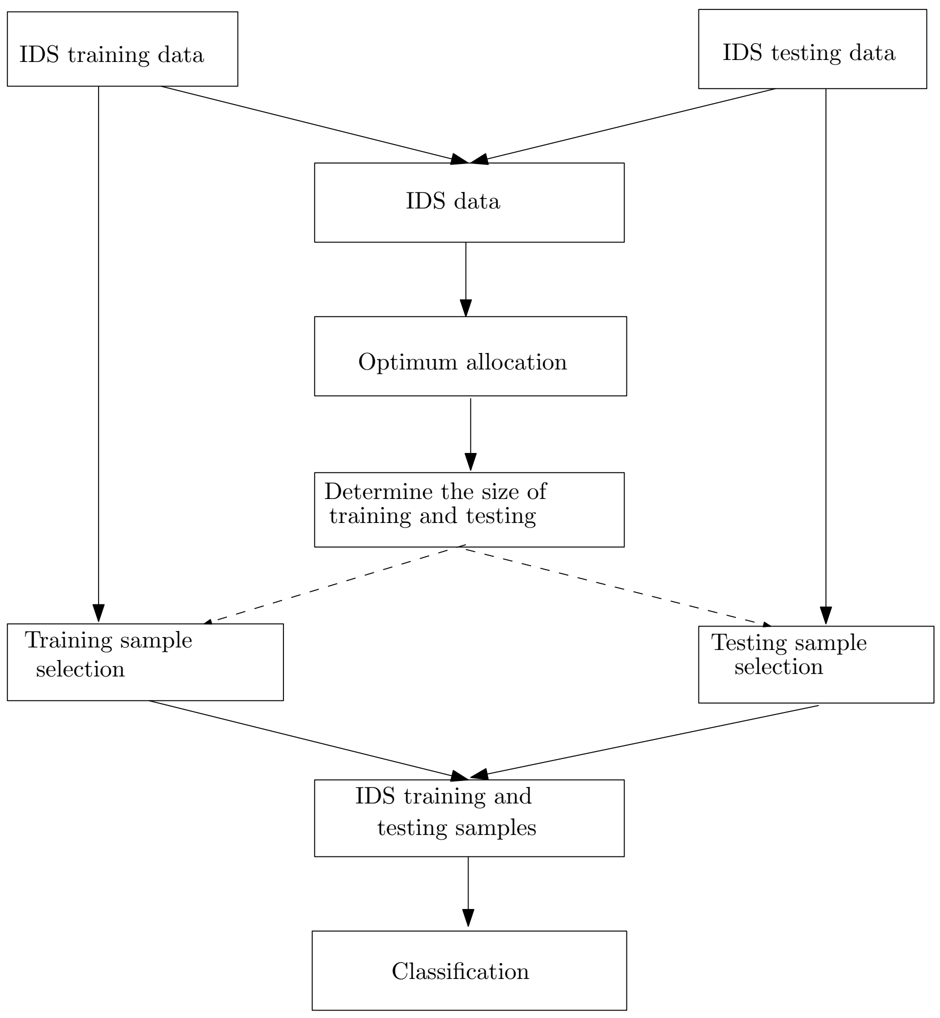
IDS training data
IDS testing data
IDS data
Optimum allocation
Determine the size of training and testing
Training sample selection
Testing sample selection
IDS training and testing samples
Classification

Figure 1: General architecture of the proposed methodology 1 (OA-LS-SVM 1)

posed OA-LS-SVM approach and the other methods in the literature are as follows: First, we develop an allocation scheme that determines the size of training and testing depending on the variability of the data. In addition, the proposed approach easily captures large datasets and can be used as a general framework for classification in pattern recognition. Secondly, it selects samples from each sub-group of training and testing depending on the variability providing a reliable means of data reduction. Finally, the proposed method can easily be implemented for incremental large datasets.

3 Proposed Methodology

In this paper, we propose a new algorithm of the optimum allocation-based least square support vector machine (OA-LS-SVM) for IDS. This section describes the methodology of

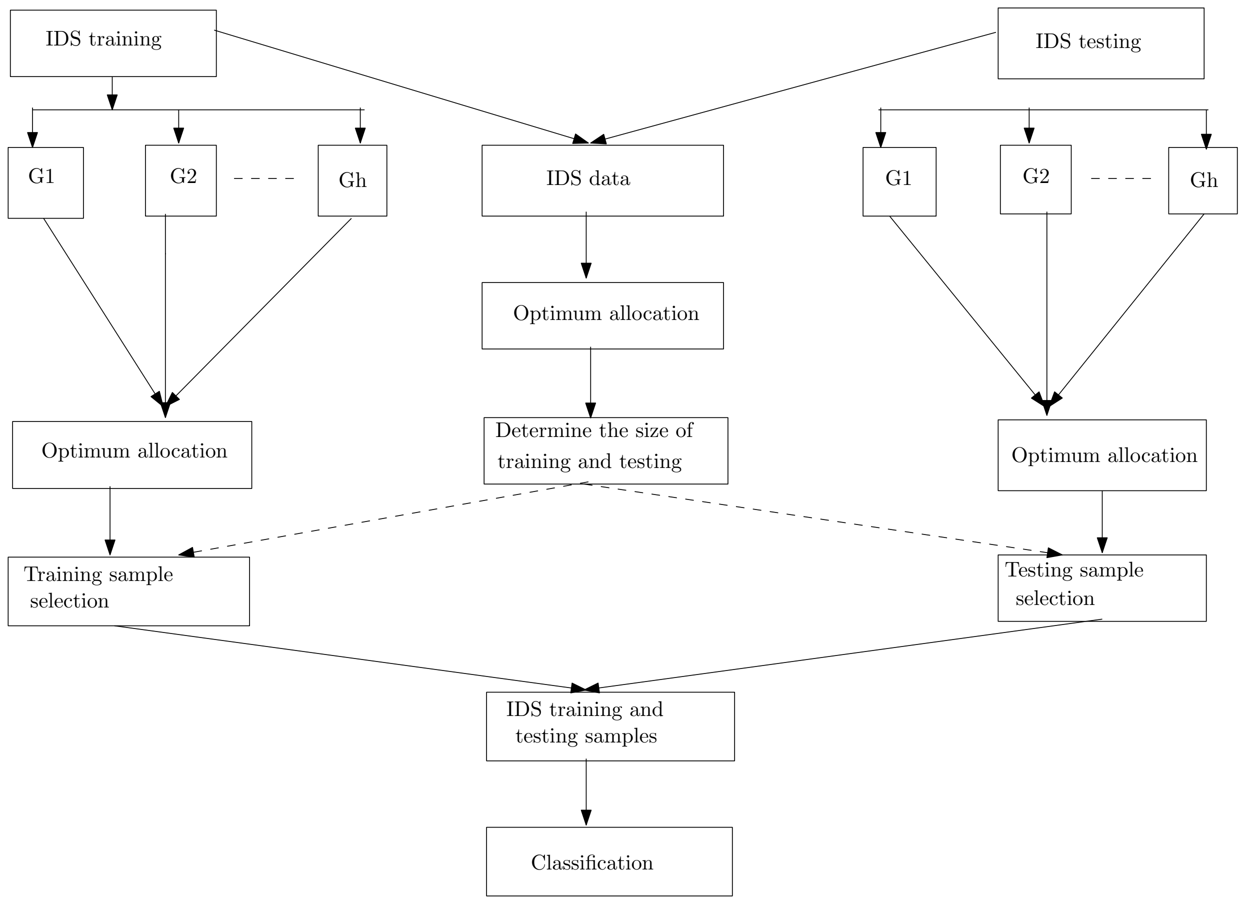

Figure 2: General architecture of the proposed methodology 2 (OA-LS-SVM 2)

the proposed algorithm both for static (i.e., the entire dataset is assumed to be available at the time of release) and for incremental datasets.

3.1 Sample Size Determination

Sample size determination is the act of choosing the number of observations to include in a statistical sample. The size of a sample is calculated on how many samples are needed in order to get results that reflect the target population as preciously as needed. In this paper we determine the required sample size in estimating population proportion by using the following popular formula as described in [2, 8, 44, 45].

$$n_0 = \frac{z^2 \star p \star q}{d^2} \tag{3.1}$$

where, n_0 = desired sample size; z = standard normal deviate (Z-value) for desired confidence level (e.g. 1.96 for 95% confidence level and 1.645 for 99% confidence level);

p=assumed proportion in the target population estimated to have a particular characteristic; q=1-p; and d= degree of accuracy desired in the estimated proportion (e.g., d=0.01 for 99-100% confidence interval).

If $\frac{n_0}{N}$ is negligible (i.e, if the population size, N is very large), n_0 is a satisfactory approximation to sample size, n. If not (i.e., N is finite and small compared to n_0), the sample size n is obtained as

$$n = \frac{n_0}{1 + \frac{(n_0 - 1)}{N}} \tag{3.2}$$

If the estimator p is not known, 0.50 (50%) is used, because for given values of z and d, it produces the largest sample size. In this paper, we use $p = 0.50$ as sample size will be then maximum.

3.2 Methodology for Static Data

The algorithm first selects a representative sample from the training and testing dataset and then the selected sample will be used as an input set of LS-SVM. Depending on the selection process, the algorithm consists of two types, namely

- OA-LS-SVM 1: After determining the sizes of training and testing using OA scheme, select the training and testing representative samples directly from the respective training and testing datasets.

- OA-LS-SVM 2: After determining the sizes of training and testing using OA scheme, the training and testing datasets are divided into some predetermined sub-groups. OA scheme will be used to determine the size of training and testing from each subgroup such that the sum of theses sizes are equal to n. Select the required representative sample from each subgroup of training and testing.

The general architecture of the proposed OA based on LS-SVM (OA-LS-SVM 1) is shown in Figure 1. As shown in Figure 1, the training and testing data are combined together. Then we use the optimum allocation (OA) scheme to determine the size of training and testing. Then the training and testing samples are selected directly from the respective training and testing sets. The dotted lines shows the required size of the training and testing. Then the classification technique will be used to detect the intrusion. Here we use LS-SVM to detect different attacks in IDS.

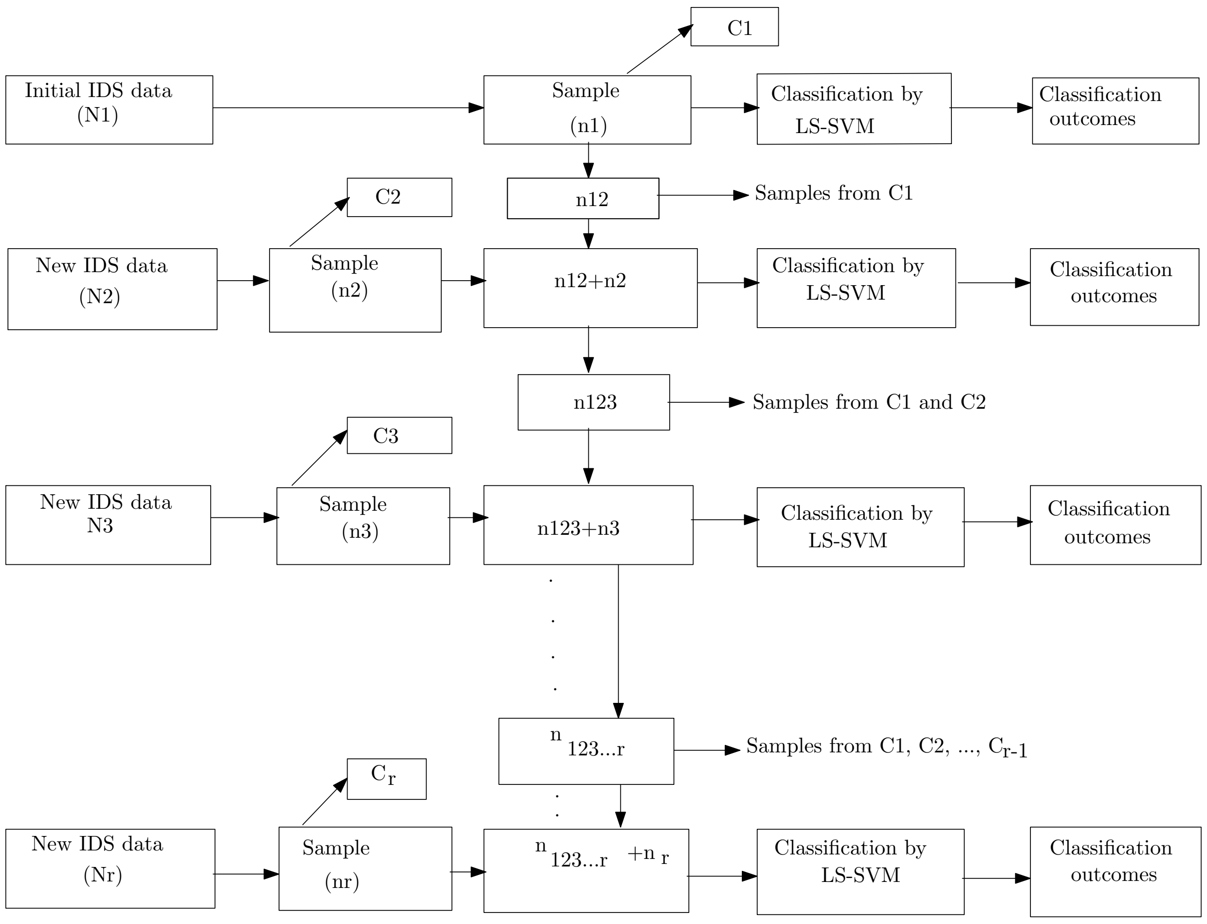

Figure 3: Block diagram of the proposed methodology for IDS in incremental datasets

On the other hand, the architecture of the proposed OA-LS-SVM 2 is ahown in Figure 2. As we can see from from Figure 2, after determining the size of training and testing, the training and testing data are divided some subgroups of arbitrary instances $(G_1, G_2, ..., G_k)$. The OA scheme is again applied to determine the size of training and testing from each subgroup. Then samples are selected from each subgroup both for training and testing such that the sum of theses sizes is equal to n. Finally LS-SVM will be used to these samples to detect different attacks in IDS.

3.3 Methodology for Incremental Data

The block diagram of the proposed OA method based LS-SVM for incremental IDS classification is shown in Figure 3. The first block of first row and first column is the input of initial IDS data and the data are continually increasing as shown in the first column of Figure 3. For the initial data, a representative sample is taken by using equation 3.2 in subsection 3.1. The obtained samples (second column of first row) is used for classification through the LS-SVM classifier in the third block of first row. For each increment, the sample selection consists of two steps. In the first step, the proposed approach employs a technique to extract representative samples from the new incremental data and in the second step a sub-samples is taken from each previously selected samples. Then the extracted samples are used as the inputs to the LS-SVM classifier. Thus for incremental dataset, the two steps are as follows:

- A representative sample from the new data. The sample size is determined using equation 3.2 in subsection 3.1.
- A representative sample from previously selected samples. Suppose that previously selected samples are considered as clusters, namely $C_1, C_2, ..., C_{r-1}$. Therefore, the total sample size necessary for the r^{th} incremental data is

  $$n_{123...r} + n_r \tag{3.1}$$

  where, n_r is the sample size required for r^{th} incremental data; $n_{123...r} = \sum_{i=1}^{r-1} n(i)$, $r = 2, 3, ...$; and $n(i)$ is the required sample size for i^{th} cluster.

  More specifically, $n_{12} = n(1)$;

  $n_{123} = n(1) + n(2)$;

  $n_{123...r} = n(1) + n(2) + ... + n(r-1)$;

3.4 Optimum Allocation Scheme

Suppose that the whole dataset consists of h subgroups. The variability of the observations within the subgroups is an important consideration in the allocation of sample sizes: the more homogeneous the subgroups are made, the greater will be the precision of this grouping process. Of course, the precision of the grouping sample largely depends on the choice of

Table 2: Size of training and testing for different pairs of attacks using OA scheme

	Training		Testing	
DOS vs NORMAL	DOS	NORMAL	DOS	NORMAL
All features	9599	9345	6608	5709
Selected features	9580	9278	6627	5776
U2R vs NORMAL	U2R	NORMAL	U2R	NORMAL
All features	52	9345	68	5709
Selected features	52	9052	68	6002
R2L vs NORMAL	R2L	NORMAL	R2L	NORMAL
All features	1126	9345	7398	5709
Selected features	1066	9278	7458	5776
PROBE vs NORMAL	PROBE	NORMAL	PROBE	NORMAL
All features	2643	9345	2883	5709
Selected features	2643	9278	2883	5776

Table 3: NORMAL and DOS (all features)

		Precision (%)	Recall (%)	F-Value (%)	Train (sec.)	Test (%)
OA-LS-SVM 1	Best	99.92	97.58	98.67		
	Avg.	99.85	97.31	**98.56**	79.36	5.98
	Worst	99.78	96.95	98.37		
OA-LS-SVM 2	Best	99.89	97.64	98.74		
	Avg.	99.86	**97.33**	98.50	78.49	5.93
	Worst	99.83	97.07	98.39		
CRF	Best	99.82	97.11	98.43		
	Avg.	99.78	97.05	98.10	256.11	64.42
	Worst	99.75	96.99	98.37		
Naïve Bays	Best	99.40	97.00	98.20		
	Avg.	99.32	97.00	98.17	1.79	26.28
	Worst	99.30	97.00	98.10		
Decision Trees	Best	99.90	97.20	98.60		
	Avg.	**99.90**	97.00	98.46	6.09	9.04
	Worst	99.90	96.70	98.30		

the sample size, n which can be determined by using the equation 3.2 in subsection 3.1. We would like to select sample from each subgroups such that the variance in the grouping process is minimum.

Let y_{ijl} is the value of l^{th} unit of the j^{th} variable in the i^{th} subgroup in sample; $i = 1, 2, ..., h; j = 1, 2, ..., k$; and $l = 1, 2, ..., n_i$; n_i is the sample size of i^{th} subgroup. Y_{ijl} is the corresponding value in the population; $l = 1, 2, ..., N_i$.

In order to find out the variability of mean in this grouping process, we assume that the samples are drawn independently in different subgroups and the sample mean is an unbiased estimator of population mean $\bar{Y}$. The mean during the grouping process is

$$\bar{y} = \frac{\sum_{i=1}^{h}\sum_{j=1}^{k}\sum_{l=1}^{n_i} y_{ijl}}{n_1k + n_2k + ... + n_hk} = \frac{\sum_{i=1}^{h}\sum_{j=1}^{k} n_i\bar{y_{ij}}}{nk}$$

where, $\bar{y_{ij}}$ is the sample mean of the j^{th} variable in the i^{th} subgroup. Similarly the corresponding mean from population is

Table 4: NORMAL and DOS (selected features)

		Precision (%)	Recall (%)	F-Value (%)	Train (sec.)	Test (%)
OA-LS-SVM 1	Best	99.74	97.30	98.50	75.096	5.61
	Avg.	99.66	**97.07**	98.34		
	Worst	99.57	96.89	98.26		
OA-LS-SVM 2	Best	99.66	97.31	98.47	77.02	5.97
	Avg.	99.60	96.82	98.19		
	Worst	99.52	96.50	98.02		
Layered CRF	Best	99.99	97.12	98.53	26.59	15.17
	Avg.	**99.98**	97.05	98.50		
	Worst	99.97	97.01	98.48		
Layered Naïve Bayes	Best	99.40	97.00	98.20	0.68	6.50
	Avg.	99.39	97.00	98.19		
	Worst	99.30	97.00	98.10		
Layered Decision Trees	Best	99.90	97.30	98.60	1.31	3.87
	Avg.	99.90	97.10	**98.50**		
	Worst	99.90	96.00	98.40		

$$\bar{Y} = \frac{\sum_{i=1}^{h}\sum_{j=1}^{k}\sum_{l=1}^{N_i} Y_{ijl}}{n_1k + n_2k + ... + n_hk} = \frac{\sum_{i=1}^{h}\sum_{j=1}^{k} N_i\bar{Y_{ij}}}{nk}$$

Thus, $\bar{y} - \bar{Y} = \frac{1}{k}[\sum_{i=1}^{h}\sum_{j=1}^{k}\frac{N_i}{N}(\bar{y_{ij}} - \bar{Y_{ij}})]$, assuming the sampling fraction is the same in all subgroups, *i.e.*, $\frac{n_i}{n} = \frac{N_i}{N}$, where, $n = n_1 + n_2 + ... + n_h$ and $N = N_1 + N_2 + ... + N_h$. Therefore the variability of the mean during the grouping process is

$$\begin{aligned} V(\bar{y}) &= E[\bar{y} - E(\bar{y})] = E[\bar{y} - \bar{Y}] \\ &= \frac{1}{k^2}E[\sum_{i=1}^{h}\sum_{j=1}^{k}\frac{N_i}{N}(\bar{y_{ij}} - \bar{Y_{ij}})] \\ &= \frac{1}{k^2}[\sum_{i=1}^{h}\sum_{j=1}^{k}\frac{N_i^2}{N^2}E(\bar{y_{ij}} - \bar{Y_{ij}})] \\ &= \frac{1}{k^2}\sum_{i=1}^{h}\sum_{j=1}^{k}\frac{N_i^2}{N^2}V(\bar{y_{ij}}) \qquad (3.1) \end{aligned}$$

Here, $\bar{y_{ij}}$ is the mean of the simple random sample in the j^{th} variable of the i^{th} subgroup, whose variance is given by [2]

$$V(\bar{y_{ij}}) = \frac{N_i - n_i}{N_i}\frac{s_{ij}^2}{n_i}$$

By substitution of this value in equation 3.1, we obtain

$$V(\bar{y}) = \frac{1}{k^2}\sum_{i=1}^{h}\sum_{j=1}^{k}\frac{N_i^2}{N^2}\frac{N_i - n_i}{N_i}\frac{s_{ij}^2}{n_i} \qquad (3.2)$$

where, N_i is the size of i^{th} subgroup; n_i is the required sample taken from i^{th} subgroup; s_{ij} is the standard deviation of the j^{th} variable of i^{th} subgroup; n is the total sample size in the grouping process. Specifically, $n = n_1 + n_2 + ... + n_h$ and $N = N_1 + N_2 + ... + N_h$.

Our problem here is to see how a given total sample size, n, should be allocated among different subgroups so that the grouping estimator, $\bar{y}$ will have the smallest possible variability. Formally, the problem is to determine $n_1, n_2, ..., n_h$ so as to minimize, $V(\bar{y})$, subject to the constraint that the total size equals $n = n_1 + n_2 + ... + n_h$. This is equal to minimizing the function

$$\begin{aligned} \psi &= V(\bar{y}) + \lambda(\sum_{i=1}^{h} n_i - n) \\ &= \frac{1}{k^2} \sum_{i=1}^{h} \sum_{j=1}^{k} \frac{N_i^2}{N^2} \frac{N_i - n_i}{N_i} \frac{s_{ij}^2}{n_i} + \lambda(\sum_{i=1}^{h} n_i - n) \end{aligned} \tag{3.3}$$

For n_i, λ being an unknown Lagrange's multiplier. For an extremum of the function, we have $\frac{\delta\psi}{\delta n_i} = 0, \text{and} \frac{\delta^2\psi}{\delta n_i^2} > 0$. Now differentiating the function ψ with respect to n_i and equating the derivative to 0, we have

$$\frac{\delta\psi}{\delta n_i} = -\frac{1}{k^2} \sum_{i=1}^{h} \sum_{j=1}^{k} \frac{N_i - n_i}{N_i} \frac{s_{ij}^2}{n_i} + \sum_{i=1}^{h} \lambda = 0$$

$$\Rightarrow n_i = \frac{N_i}{Nk\sqrt{\lambda}} \sqrt{\sum_{j=1}^{k} s_{ij}^2} \tag{3.4}$$

Summing, $\sum_{i=1}^{h} n_i = n = \frac{1}{kN\sqrt{\lambda}} \sum_{i=1}^{h} (N_i \sqrt{\sum_{j=1}^{k} s_{ij}^2})$. Therefore, $\sqrt{\lambda} = \frac{\sum_{i=1}^{h} (N_i \sqrt{\sum_{j=1}^{k} s_{ij}^2})}{kNn}$ and putting the value of $\sqrt{\lambda}$ in equation 3.4, we have

$$n_i = \frac{N_i \sqrt{\sum_{j=1}^{k} s_{ij}^2}}{\sum_{i=1}^{h} (N_i \sqrt{\sum_{j=1}^{k} s_{ij}^2})} \times n \tag{3.5}$$

For incremental dataset, Let y_{ijl} be the value of l^{th} unit of the j^{th} variable in the i^{th} cluster in sample; $i = 1, 2, ..., r-1; j = 1, 2, ..., k$; and $l = 1, 2, ..., n(i); n(i)$ = sample size from i^{th} cluster. Then according to the derivation above, the required sample size $n(i)$ for i^{th} cluster is

$$n(i) = \frac{n_i \sqrt{\sum_{j=1}^{k} s_{ij}^2}}{\sum_{i=1}^{r-1} (n_i \sqrt{\sum_{j=1}^{k} s_{ij}^2})} \times n_{123...r} \tag{3.6}$$

where, n_i is the size of the i^{th} cluster; s_{ij}^2 is the variance of j^{th} variable of i^{th} cluster, and $n_{123...r}$ is the total required size of sample from previously selected clusters.

Table 5: NORMAL and U2R (all features)

		Precision (%)	Recall (%)	F-Value (%)	Train (sec.)	Test (%)
OA-LS-SVM 1	Best	96.88	48.53	62.00	12.04	5.30
	Avg.	85.47	42.11	**57.14**		
	Worst	82.76	35.29	49.48		
OA-LS-SVM 2	Best	100	39.71	56.85	11.68	5.97
	Avg.	**95.04**	38.24	54.51		
	Worst	89.29	38.76	52.08		
CRF	Best	58.62	60.29	56.74	8.35	13.45
	Avg.	52.16	55.02	53.44		
	Worst	47.30	50.00	49.30		
Naïve Bays	Best	5.30	91.20	10.00	0.31	5.90
	Avg.	3.94	**85.88**	7.54		
	Worst	3.20	82.40	6.20		
Decision Trees	Best	24.80	63.20	34.90	0.37	2.22
	Avg.	12.93	57.49	20.42		
	Worst	6.30	51.50	11.20		

Table 6: NORMAL and U2R (selected features)

		Precision (%)	Recall (%)	F-Value (%)	Train (sec.)	Test (%)
OA-LS-SVM 1	Best	100	39.71	55.67	10.62	1.19
	Avg.	**92.74**	38.09	**53.84**		
	Worst	87.10	33.83	50.00		
OA-LS-SVM 2	Best	100	39.71	56.85	10.68	1.20
	Avg.	92.17	36.44	52.17		
	Worst	86.21	30.88	46.15		
Layered CRF	Best	58.62	60.29	56.74	0.85	2.67
	Avg.	52.16	55.02	53.44		
	Worst	47.30	50.00	49.30		
Layered Naïve Bayes	Best	5.30	91.20	10.00	0.25	1.83
	Avg.	3.94	**85.88**	7.54		
	Worst	3.20	82.40	6.20		
Layered Decision Trees	Best	24.80	63.20	34.90	0.29	0.93
	Avg.	12.93	57.49	20.42		
	Worst	6.30	51.50	11.20		

3.5 Least square support vector machine (LS-SVM) for binary classification

Recently LS-SVMs are becoming increasingly popular as a powerful tool for data classification and function estimation. The LS-SVM was originally proposed by Suykens and Vandewalle [26] and corresponds to a modified version of a support vector machine (SVM) [51]. The LS-SVM solves a set of linear equations instead of a quadratic programming problem and all training points are used to model the LS-SVM. In this paper, for the detection of intrusions in incremental datasets, an LS-SVM is used as a detector. The extracted samples obtained by the OA approach are the input to the LS-SVM. The concept of the LS-SVM is briefly introduced as follows [26]:

Consider a training set $\{x_i, y_i\}_{i=1,2,\ldots,N}$ where x_i is the ith input features vector of d-dimension, and y_i is the class label of x_i, which is either $+1$ or -1. In the feature space, the classification function can be described as

$$y(x) = \text{sign}[w^T\phi(x) + b] \tag{3.1}$$

where w is the weight vector, b is the bias term and $\phi(x)$ is nonlinear function, which is not explicitly constructed, maps the input into higher dimensional feature space (can be infinite dimension) [26]. The weight vector, w, and the bias term, b, need to be determined. In order to obtain w and b, the optimization problem to be solved is as follows

$$\text{Min}\, J(w, b, e) = \frac{1}{2}w^T w + \frac{1}{2}\gamma\sum_{i=1}^{N} e_i^2 \tag{3.2}$$

subject to the equality constraint

$$y_i[w^T\phi(x_i) + b] = 1 - e_i, i = 1, 2, ..., N \tag{3.3}$$

Here γ is the regularization parameter, e_i is the classification error variable and J is the cost function which minimizes the classification error. The Lagrangian can be defined for Equation 3.2 as

$$\begin{aligned} L(w, b, e; \alpha) = J(w, b, e) - \sum_{i=1}^{N} \alpha_i\{y_i[w^T\phi(x_i) + b] \\ - 1 + e_i\} \end{aligned} \tag{3.4}$$

where the $\alpha_i (i = 1, 2, ..., N)$ denote Lagrange multipliers. The solution of Equation 3.4 can be obtained by partially differentiating L with respect to w, b, e_i, α_i and considering the resulting equations minimizing to zero. The detailed derivation is available in the reference [26]. After solving Equation 3.4, the LS-SVM classifier can be obtained as

$$y(x) = \text{sign}(\sum_{i=1}^{N} y_i\alpha_i K(x, x_i) + b) \tag{3.5}$$

This paper uses the radial basis function (RBF) as the kernal function, which is defined [26] as

$$K(x, x_i) = \phi(x)^T\phi(x_i) = \exp(\frac{-(||x - x_i||)^2}{2\sigma^2})$$

where σ is the bandwidth parameter.

Table 7: NORMAL and R2L (all features)

		Precision (%)	Recall (%)	F-Value (%)	Train (sec.)	Test (%)
OA-LS-SVM 1	Best	82.82	72.98	77.46	15.62	3.42
	Avg.	82.37	**71.48**	76.54		
	Worst	81.93	69.38	75.13		
OA-LS-SVM 2	Best	84.15	72.15	77.65	15.53	3.40
	Avg.	**83.45**	69.88	**76.93**		
	Worst	82.40	66.38	76.09		
CRF	Best	93.67	16.81	28.42	17.16	17.16
	Avg.	92.35	15.10	25.94		
	Worst	90.54	12.42	21.89		
Naïve Bays	Best	74.10	7.40	13.40	0.38	7.33
	Avg.	70.03	6.63	12.12		
	Worst	61.30	5.40	10.00		
Decision Trees	Best	98.30	37.10	53.20	0.60	2.75
	Avg.	84.68	23.29	35.62		
	Worst	63.70	10.40	18.30		

Table 8: NORMAL and R2L (selected features)

		Precision (%)	Recall (%)	F-Value (%)	Train (sec.)	Test (%)
OA-LS-SVM 1	Best	81.00	70.00	75.60	14.42	3.29
	Avg.	80.05	68.97	74.19		
	Worst	79.60	66.31	72.60		
OA-LS-SVM 2	Best	83.85	74.48	78.84	15.36	3.50
	Avg.	83.12	**70.71**	**76.40**		
	Worst	81.11	68.22	74.60		
Layered CRF	Best	95.84	31.67	47.52	5.30	5.96
	Avg.	**94.70**	27.08	42.08		
	Worst	91.37	24.98	39.23		
Layered Naïve Bayes	Best	88.30	7.20	13.30	0.31	2.99
	Avg.	81.81	6.47	11.98		
	Worst	78.20	4.10	7.80		
Layered Decision Trees	Best	89.70	14.50	24.90	0.036	1.53
	Avg.	85.48	10.39	18.43		
	Worst	78.80	7.30	13.50		

3.6 Multiclass LS-SVM Classifiers

Multiclass LS-SVM is a straightforward extension of LS-SVM proposed by Suykens and Vandewalle [25]. This method is very popular in machine learning community because it is nicely deals with high dimensional data and provides good generalization properties. The method also determines the classifier architecture once kernel function and the parameters are chosen by user [51]. In this paper we employ the Multiclass with radial basis function (RBF) as a classifier. Consider a training set $\{y_k^{(i)}, x_k\}_{k=1,2,\ldots,N}^{i=1,2,\ldots,m}$ where x_i is the ith input features vector of d-dimension, and $y_k^{(i)}$ is the class level of the ith class for features k. The derivation of the multiclass LS-SVM [25] is based upon the formulation

$$\min_{w_i,b_i,e_{k,i}} \ell_{LS}^{(m)}(w_i, b_i, e_{k,i}) = \frac{1}{2}\sum_{i=1}^{m} w_i^T w_i + \gamma\frac{1}{2}\sum_{i=1}^{N}\sum_{i=1}^{m} e_{k,i}^2 \qquad (3.1)$$

subject to equality constraints;

$$y_k^{(1)}[w_1^T\phi_1(x_k)+b_1]=1-e_{k,1}, k=1,...,N$$

$$y_k^{(2)}[w_2^T\phi_2(x_k)+b_2]=1-e_{k,2}, k=1,...,N$$

$$...$$

$$y_k^{(m)}[w_m^T\phi_m(x_k)+b_m]=1-e_{k,m}, k=1,...,N$$

The Lagrangian can be defined for Equation 3.1 as

$$L^{(m)}(w_i,b_i,e_{k,i};\alpha_{k,i})=\ell_{LS}^{(m)}-\sum_{k,i}\alpha_{k,i}\{y_k^{(i)}[w_i^T\phi_i$$
$$(x_k)+b_i]-1+e_{k,i}\}$$

the discrimination of the Multiclass LS-SVM is obtained as below:

$$y_i(x)=\text{sign}\{\sum_{k=1}^{N}y_k^i\alpha_{ki}K_i(x,x_k)+b_i\}; i=1,2,...,m \tag{3.2}$$

where $y_i(x)$ is the predicted class on the basis of the input index x, b_i is the bias term, α_{ij} denote Lagrange multipliers called support values, and $K_i(x,x_k)$ is the RBF kernel defined [25] as $K_i(x,x-k)=\exp(-(\| x-x_k \|)^2/2\sigma^2)$. There are four approaches for the multiclass classification such as: versus One; One versus All; Minimal Output Coding (MOC); and Error Correcting Output Codes (ECOC). The detailed description of those coding system is available in reference [26]. In this paper, the first three classification systems are used to compare the reliability of the proposed algorithm.

Table 9: NORMAL and PROBE (all features)

		Precision (%)	Recall (%)	F-Value (%)	Train (sec.)	Test (%)
OA-LS-SVM 1	Best	98.25	92.09	93.86	21.78	2.41
	Avg.	96.16	90.08	93.01		
	Worst	93.25	88.17	92.70		
OA-LS-SVM 2	Best	98.30	94.00	95.73	22.49	2.44
	Avg.	**97.64**	90.89	**94.14**		
	Worst	96.69	89.21	93.32		
CRF	Best	84.60	89.94	86.73	200.6	14.53
	Avg.	82.53	88.06	85.21		
	Worst	80.44	86.13	83.19		
Naïve Bays	Best	73.20	97.00	83.30	1.08	6.31
	Avg.	72.26	**96.65**	82.70		
	Worst	71.20	96.30	81.90		
Decision Trees	Best	93.20	97.70	95.40	2.04	2.40
	Avg.	87.36	95.73	91.34		
	Worst	85.50	90.90	88.80		

Table 10: NORMAL and PROBE (selected features)

		Precision (%)	Recall (%)	F-Value (%)	Train (sec.)	Test (%)
OA-LS-SVM 1	Best	98.70	77.80	86.85	22.66	2.56
	Avg.	98.30	77.28	86.50		
	Worst	97.72	76.38	86.12		
OA-LS-SVM 2	Best	98.94	77.38	86.84	22.69	2.58
	Avg.	**98.43**	77.15	86.50		
	Worst	98.06	76.90	86.31		
Layered CRF	Best	89.72	98.03	93.68	6.91	2.04
	Avg.	88.19	**97.82**	**92.73**		
	Worst	82.92	96.48	89.82		
Layered Naïve Bayes	Best	78.80	21.30	33.60	0.45	1.13
	Avg.	77.23	19.57	31.22		
	Worst	74.70	17.00	27.70		
Layered Decision Trees	Best	87.50	97.70	92.30	0.54	1.00
	Avg.	87.04	97.41	91.93		
	Worst	86.60	95.20	90.80		

4 Experimental results

The objective of our experiment is to investigate the recital of our approach in terms of detection accuracy and computational efficiency. In this paper the KDD 99 intrusion detection dataset [52] is used which is based on the 1998 DARPA initiative, which provides designers of intrusion detection systems (IDS) with a benchmark on which to evaluate different methodologies. To do so, a simulation is made of a factitious military network consisting of three target machines running various operating systems and services. Additional three machines are then used to spoof different IP addresses to generate traffic. Finally, there is a sniffer that records all network traffic using the TCP dump format. The total simulated period is seven weeks. Normal connections are created to profile what is expected in a military network and attacks fall into one of four categories:

- Denial of Service (DOS): Attacker tries to prevent legitimate users from using a service, e.g., syn flood;
- Remote to Local (R2L): Attacker does not have an account on the victim machine, hence tries to gain access, e.g., guessing password;
- User to Root (U2R): Attacker has local access to the victim machine and tries to gain super user privileges, e.g., various “buffer over flow” attacks;
- PROBE: Attacker tries to gain information about the target host, e.g., port scanning.

Table 11: Comparison of Results (PD)

		DOS	U2R	R2L	PROBE
OA-LS-SVM 1 (One vs One) ($\gamma = 10, \sigma^2 = 10$)	PD	97.30	14.08	9.46	88.03
	FAR	0.06	0.37	0.10	1.13
OA-LS-SVM 1 (One vs All) ($\gamma = 10, \sigma^2 = 10$)	PD	98.50	67.16	9.61	93.93
	FAR	0.023	0.18	0.032	3.17
OA-LS-SVM 1 (MOC) ($\gamma = 10, \sigma^2 = 10$)	PD	98.08	63.89	9.60	91.20
	FAR	2.00	0.22	0.39	0.67
OA-LS-SVM 2 (One vs One) ($\gamma = 10, \sigma^2 = 10$)	PD	97.86	17.65	12.60	88.32
	FAR	0.05	0.17	0.07	0.80
OA-LS-SVM 2 (One vs All) ($\gamma = 10, \sigma^2 = 10$)	PD	98.77	65.60	13.46	93.58
	FAR	0.04	0.04	0.05	3.5
OA-LS-SVM 2 (MOC) ($\gamma = 10, \sigma^2 = 10$)	PD	**98.92**	**72.06**	13.51	**94.50**
	FAR	0.74	0.15	0.30	0.60
OA-LS-SVM 1 (One vs One) ($\gamma = 10, \sigma^2 = 100$)	PD	89.33	61.47	68.98	93.97
	FAR	0.07	0.003	0.05	21.42
OA-LS-SVM 1 (One vs All) ($\gamma = 10, \sigma^2 = 100$)	PD	97.27	66.47	70.87	90.00
	FAR	0.045	0.007	0.028	17.82
OA-LS-SVM 1 (MOC) ($\gamma = 10, \sigma^2 = 100$)	PD	97.40	62.50	69.65	89.46
	FAR	.11	0.04	0.27	20.91
OA-LS-SVM 2 (One vs One) ($\gamma = 10, \sigma^2 = 100$)	PD	91.64	59.41	68.65	94.26
	FAR	0.10	0.018	0.04	18.85
OA-LS-SVM 2 (One vs All) ($\gamma = 10, \sigma^2 = 100$)	PD	97.20	66.77	69.60	90.64
	FAR	0.06	0.018	0.03	18.20
OA-LS-SVM 2 (MOC) ($\gamma = 10, \sigma^2 = 100$)	PD	97.62	67.94	**71.65**	89.65
	FAR	0.16	0.04	0.34	18.43
KDD' 99 Winner	PD	97.10	13.20	8.40	83.30
	FAR	0.30	0.003	0.005	0.60
Multi Classifier	PD	97.30	29.80	9.60	88.70
	FAR	0.40	0.40	0.10	0.40
Multi Layer Perception	PD	97.20	13.20	5.60	88.70
	FAR	0.30	0.05	0.010	0.40
Gaussian Classifier	PD	82.40	22.80	9.60	90.20
	FAR	0.90	0.50	0.10	11.30
K-Means Clustering	PD	97.30	29.80	6.40	87.60
	FAR	0.40	0.40	0.10	2.60
Nearest Cluster Algorithm	PD	97.10	2.20	3.40	88.80
	FAR	0.30	0.0006	0.010	0.50
Incremental Radial Basis Function	PD	73.00	6.10	5.90	93.20
	FAR	0.20	0.04	0.30	18.80
Leader Algorithm	PD	97.20	6.60	0.10	83.80
	FAR	0.30	0.03	0.003	0.30
Hypersphere Algorithm	PD	97.20	8.30	1.00	84.80
	FAR	0.30	0.009	0.005	0.40
Fuzzy ARTMAP	PD	97.00	6.10	3.70	77.20
	FAR	0.30	0.001	0.004	0.20
C4.5 (Decision Trees)	PD	97.00	1.80	4.60	80.80
	FAR	0.30	0.002	0.005	0.70
Nearest Neighbour with Principle Component Analysis (4 axis)	PD	97.32	64.04	2.51	86.13
	FAR	0.23	0.0001	0.001	.27
Decision Trees with Principle Component Analysis (2 axis)	PD	97.58	7.02	0.070	70.40
	FAR	0.12	0.0001	0.030	0.85
Support Vector Machines	PD	91.60	12.00	22.00	36.65
	FAR	-	-	-	-
Layered Conditional Random Fields (all features)	PD	97.05	55.03	15.10	88.06
	FAR	-	-	-	-

4.1 Results for Static Data

The data set contains about five million connection records as the training data and about two million connection records as the test data. In our experiments, we use 10% of the total training data and 10% of the test data (with corrected labels), which are provided separately. This leads to 494,021 training and 311,029 test instances. Each in the data set represents a connection between two IP addresses, starting and at some well defined times with a well-defined protocol. Further every record is represented by 41 different features (variables). Each record represents a separate connection and is hence considered to be independent of any other record. The training data is either labeled as NORMAL or as one of the 24 different kinds of attack. These 24 attacks can be grouped into four classes; Probing, DOS, R2L, and U2R. Similarly, the test data is also labeled as either NORMAL or as one of the attacks belonging to the four attack groups. It is important to note that the test data is not from the same probability distribution as the training data, and it includes specific attack types not present in the training data. This makes the intrusion detection task more realistic [52]. In this paper, the classification by the LS-SVM is carried out in MATLAB (ver. 7.14, R2012a) using the LS-SVMlab toolbox (ver. 1.8) [53]. We perform experiments on a desktop running with Intel(R) Core(TM) i7-2600, CPU 3.40 GHz,and 8-Gbyte RAM under the same conditions.

For our results, we give the Precision, Recall, and F-Value as of [18] and not the accuracy alone as with the given data set, it is easy to achieve very high accuracy by carefully selecting the sample size. The Precision, Recall, and F-Value are defined as follows:

$$\begin{aligned}\text{Precision} &= \frac{\text{TP}}{\text{TP+FP}}\\ \text{Recall} &= \frac{\text{TP}}{\text{TP+FN}}\\ \text{F-value} &= \frac{(1+\beta^2)\times\text{Recall}\times\text{Precision}}{\beta^2\times(\text{Recall+Precision})}\end{aligned}$$

where TP, FP, and FN are the number of True Positives, False Positives, and False Negatives, respectively, and β corresponds to the relative importance of precision versus recall and is usually set to 1.

Table 12: Comparison of Results (PCD)

		DOS	U2R	R2L	PROBE
OA-LS-SVM 1 (One vs One) ($\gamma = 10, \sigma^2 = 10$)	PCD	97.22	3.20	7.81	70.20
	FAR	0.06	0.37	0.10	1.13
OA-LS-SVM 1 (One vs All) ($\gamma = 10, \sigma^2 = 10$)	PCD	97.24	3.92	7.48	69.27
	FAR	0.023	0.18	0.032	3.17
OA-LS-SVM 1 (MOC) ($\gamma = 10, \sigma^2 = 10$)	PCD	97.80	17.35	7.05	71.31
	FAR	2.00	0.22	0.39	0.67
OA-LS-SVM 2 (One vs One) ($\gamma = 10, \sigma^2 = 10$)	PCD	97.46	5.59	**11.19**	**77.76**
	FAR	0.05	0.17	0.07	0.80
OA-LS-SVM 2 (One vs All) ($\gamma = 10, \sigma^2 = 10$)	PCD	97.55	4.78	11.14	75.55
	FAR	0.04	0.04	0.05	3.5
OA-LS-SVM 2 (MOC) ($\gamma = 10, \sigma^2 = 10$)	PCD	**98.24**	16.18	10.97	73.55
	FAR	0.74	0.15	0.30	0.60
OA-LS-SVM 1 (One vs One) ($\gamma = 10, \sigma^2 = 100$)	PCD	89.20	37.24	5.80	84.29
	FAR	0.07	0.003	0.05	21.42
OA-LS-SVM 1 (One vs All) ($\gamma = 10, \sigma^2 = 100$)	PCD	87.60	**37.35**	2.65	75.67
	FAR	0.045	0.007	0.028	17.82
OA-LS-SVM 1 (MOC) ($\gamma = 10, \sigma^2 = 100$)	PCD	97.30	32.72	3.28	75.10
	FAR	.11	0.04	0.27	20.91
OA-LS-SVM 2 (One vs One) ($\gamma = 10, \sigma^2 = 100$)	PCD	91.13	36.47	8.05	**86.36**
	FAR	0.10	0.018	0.04	18.85
OA-LS-SVM 2 (One vs All) ($\gamma = 10, \sigma^2 = 100$)	PCD	86.52	35.30	3.38	76.82
	FAR	0.06	0.018	0.03	18.20
OA-LS-SVM 2 (MOC) ($\gamma = 10, \sigma^2 = 100$)	PCD	97.27	34.21	4.42	76.28
	FAR	0.16	0.04	0.34	18.43
Bernhard [47]	PCD	97.10	13.2	8.4	83.30
	FAR	-	-	-	-
Y.Liu [48]	PCD	56.00	66.00	78.00	44.000
	FAR	-	-	-	-
Kayacik [49]	PCD	95.10	10.00	9.9	64.3
	FAR	-	-	-	-
Ambwani [50]	PCD	96.8	4.2	5.3	75
	FAR	-	-	-	-

We divide the training and testing data into different groups; DOS, U2R, R2L, PROBE and NORMAL. The LS-SVM classifier is trained with the training set and performances are assessed with the testing set for different pairs of the two-class data. For detecting DOS attacks, we train and test the system with DOS and NORMAL data only. Similarly, for detecting U2R attacks, we train and test the system with U2R and NORMAL data only. Thus we have four pairs of binary class which are as follows:

DOS versus NORMAL

U2R versus NORMAL

R2L versus NORMAL

PROBE versus NORMAL

We combine the training and testing for each of the pairs and determine the total sample size required that describe the characteristics of training and testing. As mentioned before we use equation 3.2 of subsection 3.1 to determine the sizes using 99% confidence level and 99%-100% confidence interval. The determination of training and testing set are discussed in Section 3.2. As indicated by Gupta et.al [18] selected features may be useful for detecting certain type of attack. Thus we compare our results with the selected features as well. Table 2 presents the required size of training and testing for all features as well as for selected features using the OA scheme as described in Section 3.4. It should be noted that the training and testing for U2R attack is very small and thus we consider all of these instances for classification. For PROBE attack we determine the size training and testing considering each feature separately and then take the average. We perform 10 experiments for each attack class by randomly selecting data corresponding to that attack class and normal data only and recorded best, average and worst value. We compare our results with Conditional Random Field (CRF), Naïve Bayes and Decision Trees. The Precision, Recall and F-value for these algorithms for the KDD 99 dataset are quoted from [18].

4.1.1 Detecting DOS attack

As discussed in Section 3.2, using OA scheme we randomly select 9345 NORMAL records and 9599 DOS records from the training data as the training for detecting DOS attacks.

Using the same scheme, we randomly select 5709 NORMAL and 6608 records from the test data for testing. Hence, we have 18,944 training instances and 12,317 testing instances. Table 3 gives the results for the experiment. In this paper, the stability of performance of the proposed OA-LS-SVM classifier is assessed based on different statistical measurements, such as Precision, Recall and F value. The RBF kernel function is employed for the LS-SVM as an optimal kernel function over different kernel functions that were tested. The LS-SVM has two important parameters γ and σ^2, which should be appropriately chosen for achieving the desired performance. In order to obtain the best results, the LS-SVM is trained with different combinations of the parameters γ and σ^2. For detecting DOS attack, the optimal detection results are obtained for OA-LS-SVM 1 and OA-LS-SVM 2 as $\gamma = 1000$ and $\sigma^2 = 10$ and $\gamma = 10$ and $\sigma^2 = 10$ respectively. We observe that the OA-LS-SVM 2 takes only 5.93 seconds to label all the test instances. We further observe that F-value is higher for OA-LS-SVM 1, Recall is higher for OA-LS-SVM 2 and Precision is higher for Decision Trees. Thus from this experiment, we conclude that the OA-LS-SVMs are better choice for detecting DOS attack. The highest average values of Precision, Recall and F-value are shown in bold face. We also perform experiments of using feature selection and the results are given in Table 4. For OA-LS-SVM 1 and OA-LS-SVM 2, the optimum parameters are selected as $\gamma = 1000$ and $\sigma^2 = 10$. As we can see from Table 4, the precision and F-value are higher for Layered Conditional Random Field whereas Recall is higher for OA-LS-SVM 1. It should be noted that the feature selection is a subjective process and may not uniformly perform better for other datasets. In real scenario, it is ideal to deal with all features. Thus although using feature selection, Layered CRFs may be better choice for DOS attack in this particular dataset but there is no guarantee that it will perform better for any other datasets.

4.1.2 Detecting U2R attack

Using the OA scheme, we randomly select 9345 NORMAL records as the training and 5709 records for the testing for detecting U2R attacks. We use all the training and testing records of U2R as the number of instances are small. Hence, we have 9,397 training instances and

5,777 testing instances. Table 5 gives the results for the experiment. For detecting U2R attack, the optimal detection results are obtained for OA-LS-SVM 1 and OA-LS-SVM 2 as $\gamma = 1000$ and $\sigma^2 = 1000$ and $\gamma = 10$ and $\sigma^2 = 1000$ respectively. We observe that the OA-LS-SVM 1 takes 5.30 seconds to label all the test instances. We further observe that the highest precision and F-value occurs for OA-LS-SVM 2 and for OA-LS-SVM 1 respectively whereas Naïve Bays has the highest Recall. CRF is not at all suitable for detecting U2R attack. Similar as DOS, the experimental results show that the OA-LS-SVMs are better choice for detecting U2R attack. Using the feature selection, the experimental results are given in Table 6. In this scenario, the parameters for both OA-LS-SVM are chosen as $\gamma = 100$ and $\sigma^2 = 1000$. Similar to all features, the proposed OA-LS-SVM performs better for selected features as well.

4.1.3 Detecting R2L attack

Using the OA scheme, we randomly select 9345 NORMAL records as the training and 5709 NORMAL records for the testing for detecting R2L attacks. As the training data of R2L is small, we use all the 1126 R2L instances as training and the rest 7398 R2L instances as testing. Hence, we have 10,471 training instances and 13,107 testing instances. Table 7 gives the results for the experiment. For detecting R2L attack, the optimal detection results are obtained for both OA-LS-SVM 1 and OA-LS-SVM 2 is $\gamma = 1000$ and $\sigma^2 = 100$. The testing time for OA-LS-SVM 2 is 3.40 seconds to label all the test instances. The highest precision and F-value for detecting R2L attack obtained for OA-LS-SVM 2 and the highest Recall are obtained fror OA-LS-SVM 1. The CRF, Naïve Bays and Decision Tress are not at all suitable for detecting R2L attack. Thus similar to DOS, U2R, the experimental results show that the OA-LS-SVMs are better choice for detecting R2L attack. Table 8 shows the experimental results for detecting R2L attack using selected features. The required instances for training and testing are obtained by using OA scheme. After several trials and errors, the parameters for both OA-LS-SVM are chosen as $\gamma = 1000$ and $\sigma^2 = 100$. The highest Precision are obtained for Layered CRFs but the other measures (Recall and F value) are obtained for OA-LS-SVM 2.

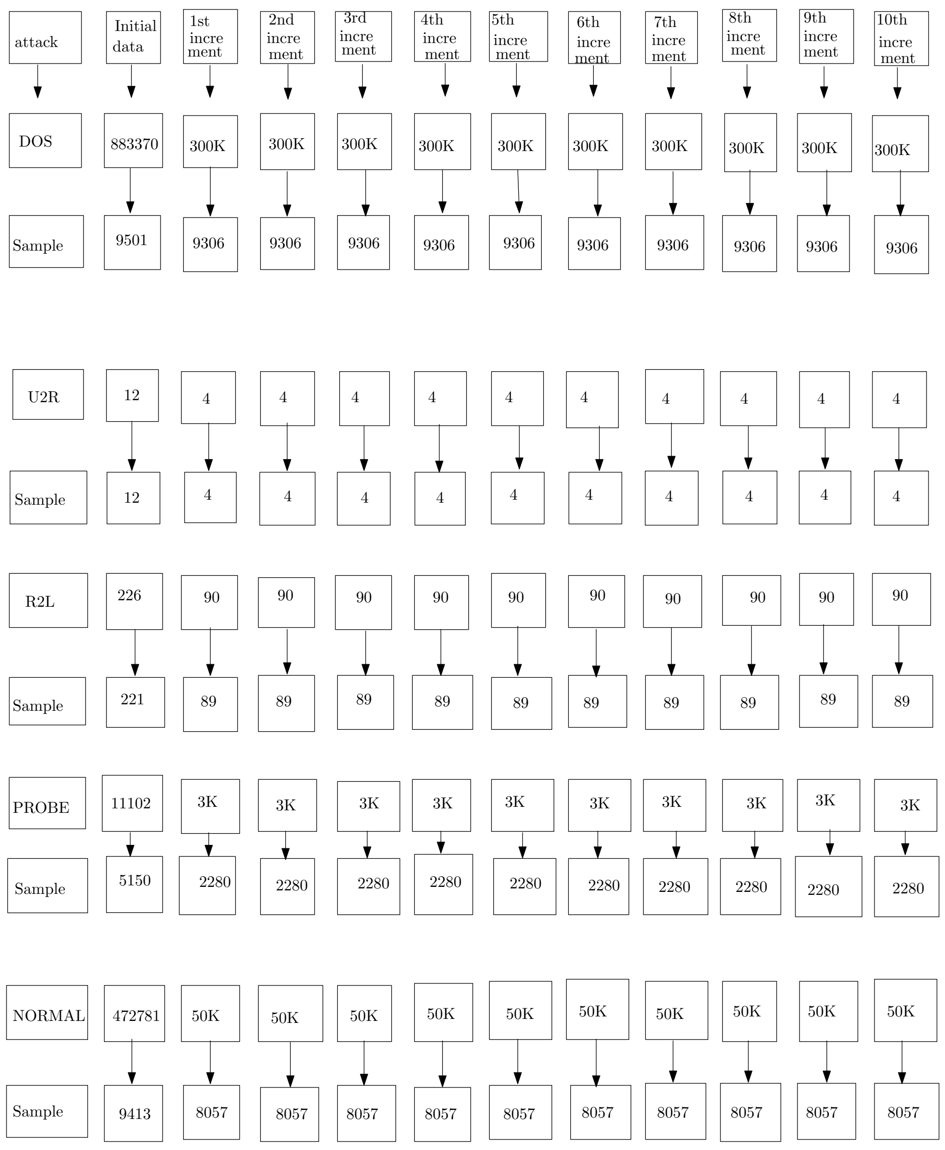

Figure 4: Block diagram of the proposed methodology for 10th incremental data

4.1.4 Detecting PROBE attack

we randomly select 2643 PROBE records as the training and 2883 PROBE records for the testing for detecting PROBE attacks. On the other hand, the respective training and testing for NORMAL records using OA scheme are 9345 and 5709. The experimental results are given in Table 9. For detecting PROBE attack, the optimal detection results are obtained for both OA-LS-SVM 1 and OA-LS-SVM 2 is $\gamma = 1000$ and $\sigma^2 = 1000$. The testing time for OA-LS-SVM 1 is 2.41 seconds to label all the test instances. The highest precision and F-value for detecting PROBE attack are found for OA-LS-SVM 2 and the highest Recall are obtained for Naïve Bays. The CRF and Decision Tress are not at all suitable for detecting PROBE attack. Thus similar to DOS, U2R and the R2L, the experimental results show that the OA-LS-SVMs are better choice for detecting R2L attack. Table 10 shows the experimental results for detecting PROBE attack using feature selection. The optimum parameters for both OA-LS-SVM in this situation are chosen as $\gamma = 10$ and $\sigma^2 = 10$. The highest Precision are obtained for the OA-LS-SVM 2 whereas the highest Recall and F-value are obtained for Layered CRFs. From the experimental results from Section 4.1.1 to 4.1.4, there is no evidence that the selected features are effective for detecting attacks. Thus for further consideration of detecting attacks, we will consider all the features in the dataset.

4.2 Multiclass Comparison for Static data

This section shows the performance analysis of the proposed OA-LS-SVM in comparison with the other algorithms. We use the same dataset for multiclass comparison. The number of training and testing records for different attack are the same as in Table 2. The test performance for a given classifier of a specific category can be determined by the computation of Probability of Correct Detection (PCD), the Probability of Detection (PD) and False alarm rate (FAR). The PCD, PD and the FAR are defined as follows:

- PCD: the number of cases that are correctly detected divided by the total number of cases in that attack category;

- PD: the number of cases that are detected as attack/attacks divided by the total number of cases in that attack category;
- FAR: the number of NORMAL cases that are detected as intrusions in a specific category divided by the total number of NORMAL cases.

There are four approaches for multiclass classification such as: One versus One; One versus All; Minimal Output Coding (MOC); and Error Correcting Output Codes (ECOC). However, ECOC does not provide new information in this particular dataset that we used in this paper to evaluate the proposed algorithm. Thus in this paper we use first three approaches for OA-LSSVM. We perform the experiments for different values of the parameters γ and σ^2 and finally present results for $\gamma = 10$, $\sigma^2 = 10$ and $\gamma = 10$ and $\sigma^2 = 100$. We repeat all experiments five times and average values are recorded in Table 12 and Table 11. Table 12 shows the results of PCD whereas and Table 11 shows the results of PD.

We compare our work with other well-known methods based on the anomaly intrusion detection principle using PD. For anomaly detection, standard technoques such as CRF, decision trees and Naïve Bays are known to perform well. However, our experiments show that the OA-LS-SVM performs far better than these techniques. The main reason for this is that the OA-LS-SVM uses the training and testing set that best describe the characteristics of whole population. Sabhnani et al. [46] present a comparative study of various classifiers when applied to the KDD'99 dataset. Bouzida et al. [37] proposed to use Principle Component Analysis (PCA), before applying a machine learning algorithm. Use of support vector machines is discussed in [17] and the idea of Layered and CRF are used in [18]. We compare our results from the results presented in these papers in Table 11. The table represents the Probability of Detection (PD) and False Alarm Rate (FAR) in percent for various methods including the KDD 99 cup winners. On the other hand, we have reported and compare the Probability of correct detection (PCD) in Table 12. In respect of PCD, the Table 12 shows that our proposed OA-LS-SVM performs better than the other methods in the literature.

Table 13: Total required samples for KDD 99 dataset in 10^{th} increment

Attack	Total size of previously selected samples	Samples from column 2	Size of 10^{th} incremental data	Samples from column 4	Total
DOS	93255	8707	300000	9306	18013
U2R	48	48	4	4	52
R2L	1022	924	90	89	1083
PROBE	25724	6993	3000	2286	9279
NORMAL	81926	8596	50000	8057	16653

Table 14: Classification accuracy and false alarm rate for DOS vs NORMAL for five repeats

Repeat	Training		Testing		CA (%)	FAR (%)
	DOS	NORMAL	DOS	NORMAL		
1	10000	10000	8012	6654	100	0
2	10000	10000	8015	6652	99.99	0.015
3	10000	10000	8012	6654	99.99	0
4	10000	10000	8014	6654	100	0
5	10000	10000	8012	6652	99.98	0

4.3 Results for Incremental Data

In this experiment, we use the whole dataset of about five millions connection records of the training data. Although there about two millions connection records of test data but they are not labeled. Thus we consider the five millions training data as the whole set and use about 50% of them as the testing for detecting intrusions. In this particular situation, we consider 10^{th} occasion. That means we have the whole dataset for 10th occasion and also the datasets that were used in previous occasions. We use equation 3.2 in subsection 3.1 for determination of required sizes in the most recent occasion as well as from previous occasions. The test performance for the classifier of a specific category is determined by the computation of classification accuracy and total False Alarm Rate (FAR) are defined as follows:

- Classification Accuracy (CA): the number of cases that are correctly detected divided by the total number of cases in that attack category, same as PCD;
- total FAR: the number of NORMAL cases that are detected as intrusions divided by the total number of NORMAL cases.

Table 15: Classification accuracy and false alarm rate for U2R vs NORMAL for five repeats

Repeat	Training		Testing		CA (%)	FAR (%)
	U2R	NORMAL	U2R	NORMAL		
1	28	10000	24	6655	99.87	0
2	28	10000	24	6653	99.96	0.015
3	28	10000	24	6653	99.96	0
4	28	10000	24	6653	99.93	0.03
5	28	10000	24	6652	100	0

Table 16: Classification accuracy and false alarm rate for R2L vs NORMAL for five repeats

Repeat	Training		Testing		CA (%)	FAR (%)
	R2L	NORMAL	R2L	NORMAL		
1	550	10000	440	6652	99.84	0.03
2	550	10000	441	6653	99.76	0.03
3	550	10000	441	6652	99.68	0
4	550	10000	430	6654	99.86	0.03
5	550	10000	425	6655	99.73	0.045

4.3.1 Samples for 10^{th} increment

Figure 4 shows the block diagram of the proposed methodology for 10^{th} incremental data. In each attack type and each increment, the data are selected on the basis of the following rule

$$N_0 + 10n = N \tag{4.1}$$

where N is the total dataset of a particular attack type, N_0 is the most recent data for that attack, and n is the size of data of that attack for each increment. The block diagram also shows the required sample sizes for most recent as well as for incremental data of each attack. The required samples in the 10th increment for each attack are given in Table 13. We use 95% confidence level and 99-100% confidence interval for selecting sizes in this situation. We assume that the samplers that were used in previous occasions are available at the final occasion.

In this paper, we investigate the potentials of applying the optimum allocation algorithm for obtaining representative samples from all dataset and these samples are used as inputs to the LS-SVM algorithm. The RBF kernal function is employed for the LS-SVM as an optimal kernel function over different kernel functions that were tested. The LS-SVM has two important parameters γ and σ^2, which should be appropriately chosen for achieving the desired performance. In order to obtain the best results, the LS-SVM is trained with different combinations of the parameters γ and σ^2. The proposed method is conducted on

Table 17: Classification accuracy and false alarm rate for PROBE vs NORMAL for five repeats

Repeat	Training		Testing		CA (%)	FAR (%)
	PROBE	NORMAL	PROBE	NORMAL		
1	5200	10000	4079	6652	99.72	0.11
2	5200	10000	4078	6653	99.80	0.015
3	5200	10000	4078	6653	99.70	0.03
4	5200	10000	4079	6654	99.69	0.12
5	5200	10000	4078	6653	99.79	0.045

Table 18: Training and Testing dataset for each attack in multiclass classification

	Training	Testing
DOS	10050	7963
U2R	28	24
R2L	565	424
PROBE	5300	3977
NORMAL	9900	6754

different pairs of two-class of KDD 99 data.

4.3.2 Detecting DOS for 10^{th} increment

As shows in Table 13 we need to select 8707 instances for DOS and 8596 for NORMAL from previously selected clusters for detecting DOS attacks. Using the OA scheme in equation 3.6, we select the samples from previously selected clusters. In addition, the respective size for the most recent datasets are 9306 and 8057 respectively. Thus for 10^{th} increment, the required sizes of DOS and NORMAL are 18313 and 16653 for detecting DOS attacks. In order to obtain the best results, the LS-SVM is trained with different combinations of the parameters γ and σ^2. For detecting DOS attack in 10^{th} increment, the optimal detection results are obtained for $\gamma = 100$ and $\sigma^2 = 100$. For consistency of the proposed approach, we repeat the experiment (DOS vs NORMAL) five time for the parameters $\gamma = 100$ and $\sigma^2 = 100$ and their results are given in Table 14. As shows in Table 14 the proposed model is very accurate in the terms of classification accuracy and FAR. The performance of the classification is due to the large number of instances in both DOS and NORMAL. That

Table 19: Classification accuracy for multiclass classification (One vs all) for five repeats

Repeat	DOS (%)	U2R (%)	R2L (%)	PROBE (%)	NORMAL (%)	Overall (%)
1	99.96	83.33	98.35	99.35	99.64	99.67
2	99.94	87.50	98.11	99.32	99.67	99.66
3	99.96	91.67	98.58	99.35	99.70	99.70
4	99.92	87.50	97.41	99.30	99.66	99.63
5	99.94	91.67	97.17	99.25	99.72	99.64

Table 20: Classification accuracy for multiclass classification (One vs One) for five repeats

Repeat	DOS (%)	U2R (%)	R2L (%)	PROBE (%)	NORMAL (%)	Overall (%)
1	99.94	87.50	96.7	99.62	99.61	99.67
2	99.99	83.33	97.88	99.65	99.67	99.75
3	99.95	79.17	97.64	99.57	99.64	99.69
4	99.97	91.67	98.35	99.74	99.69	99.78
5	99.89	91.67	97.17	99.35	99.64	99.62

means, we can expect better accuracy considering the required sample as proposed in this paper even if the size of the population is large enough.

4.3.3 Detecting U2R for 10th increment

We randomly select 16653 (8057 from 10th increment and 8596 from previously selected clusters) NORMAL records and all U2R for detecting U2R attacks. We use OA scheme to determine the sizes from previously selected clusters as described in equation 3.6. The performance of the detection is best for the parameter $\gamma = 1000$ and $\sigma^2 = 100$. For consistency of the proposed approach, we repeat the experiment (U2R vs NORMAL) five time for the parameters $\gamma = 1000$ and $\sigma^2 = 100$ and their results are given in Table 15. It shows from Table 15 that the results are consistent during repeating process.

4.3.4 Detecting R2L for 10th increment

For detecting R2L, the required sizes of R2L and NORMAL are 1083 and 16653 respectively for detecting R2L attacks of which 924 R2L records are from previously selected clusters and 89 from 10th increment. On the other hand the respective sizes of Normal records from previously selected clusters and 10th increment are 8596and 8057. Same as before, the determination of sample sizes from each of the previously selected clusters are calculated by using equation 3.6. We randomly select 9900 NORMAL records and 565 R2L records as the training for detecting R2L attacks. On the other hand, we randomly select 6754 NORMAL and 424 R2L records for testing. The best performance for detecting R2L in 10th increment is achieved for the parameters $\gamma = 100$ and $\sigma^2 = 10$. For consistency of the proposed approach, we repeat the experiment (R2L vs NORMAL) five time for the parameters $\gamma = 100$ and $\sigma^2 = 10$ and their results are given in Table 16. Table 16 shows the classification accuracies are very consistent.

4.3.5 Detecting PROBE for 10^{th} increment

Using the OA scheme as described in Section 3.3 (equation 3.6), we randomly select 10,000 NORMAL records and 5,300 PROBE records as the training for detecting PROBE attacks. In addition, we randomly select 6754 NORMAL and 3977 PROBE records for testing. The performance of the detection is best for the parameter $\gamma = 100$ and $\sigma^2 = 10$. Same as before, for consistency of the proposed approach, we repeat the experiment (PROBE vs NORMAL) five time for the parameters $\gamma = 100$ and $\sigma^2 = 10$ and their results are given in Table 17. It shows from Table 17 that the results are consistent during repeating process.

4.3.6 Multiclass Classification for the 10^{th} increment

Multiclass is an extension of binary classification. There are several approaches for multiclass LS-SVM, One versus One, One versus All, MOC and ECOC. We use first two approaches in this paper for the 10 th increment to show the classification accuracy of our proposed approach. Table 18 shows the training and testing dataset for each class in multiclass classification. The highest classification accuracy occurs at $\gamma = 1000$ and $\sigma^2 = 10$ for multiclass classification of One versus All. For consistency of the proposed approach, we repeat the multiclass experiment (One vs All) five time for the parameters $\gamma = 1000$ and $\sigma^2 = 10$ and the results are given in Table 19. Table 19 shows the classification accuracies are very consistent for multiclass LSSVM. Thus the proposed OA based LS-SVM is very effective for detecting intrusions for incremental datasets. We also validate our approach with the approaches of One versus One. The highest classification accuracy for One versus One occurs at $\gamma = 100$ and $\sigma^2 = 10$. For consistency of the proposed approach, we repeat the multiclass experiment (One vs One) five time for the parameters $\gamma = 100$ and $\sigma^2 = 10$ and for the same training and testing dataset. The results of the repeated experiment are given in Table 20.

5 Conclusion

Accurate detection of various types of attack in IDS is a complicated problem, requiring the analysis of large sets of IDS data. Representative samples from a large data set play an important role to detect intrusions in the field of network security. However the current solutions for detecting intrusions is only for static datasets. This paper proposes an IDS that can be used both for static and incremental data. The proposed IDS uses the idea of sampling and we refer to this as the optimum allocation based least square support vector machine (OA-LS-SVM). The proposed methodology is discussed and validated through KDD 99 dataset which is considered as a benchmark for testing any IDS approach. The experimental results show that the proposed method is every effective for detecting intrusions for static (i.e., the entire dataset is assumed to be available at the time of release) as well as for incremental datasets.

References

[1] J.Hu, *Host-Based Anomaly IDS.* Springer Handbook of Information and Communication Security, Springer Verlag, 2010, ISBN978-3-642-04116-7 (Print), 978-3-642-04117-4 (Online).

[2] W.G. Cochran, *Sampling Techniques.* Wiley, New York, 1977.

[3] C. Dartigue, H. I. Jang and W. Zeng, "A New Data -Mining Based Approach for Network Intrusion Detection," *Proc. 7th Annual Communication Networks and Services Research Conference,* pp. 372–377, 2009.

[4] W. Lee and S.J. Stolfo, "Data mining approaches for intrusion detection," *Proc. 7th USENIX Security Symposium* pp. 79–94, 1998.

[5] W. Lee, S. Stolfo, and K. Mok, "Mining Audit Data to Build Intrusion Detection Models," *Proc. Fourth Intl Conf. Knowledge Discovery and Data Mining (KDD 98),* pp. 66-72, 1998.

[6] N.B. Amor, S. Benferhat, and Z. Elouedi, "Naïve Bayes vs. Decision Trees in Intrusion Detection Systems, *Proc. ACM Symp. Applied Computing (SAC 04),* pp. 420-424, 2004.

[7] W. Lee, S. Stolfo, and K. Mok, "A Data Mining Framework for Building Intrusion Detection Model," *Proc. IEEE Symp. Security and Privacy (SP 99)*, pp. 120–132, 1999.

[8] M.N. Islam, *An Introduction to Sampling Methods: Theory and Applications.* Book World, Dhaka, 2007.

[9] L. Portnoy, E. Eskin, and S. Stolfo, "Intrusion Detection with Unlabeled Data Using Clustering, *Proc. ACM Workshop Data Mining Applied to Security (DMSA),* 2001.

[10] H. Shah, J. Undercoffer, and A. Joshi, "Fuzzy Clustering for Intrusion Detection, *Proc. 12th IEEE Intl Conf. Fuzzy Systems (FUZZ-IEEE 03)*, vol. 2, pp. 1274–1278, 2003.

[11] T. Abraham, *IDDM: Intrusion Detection Using Data Mining Techniques,* http://www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA392237, 2012.

[12] R. Agrawal, T. Imielinski, and A. Swami, "Mining Association Rules between Sets of Items in Large Databases, *Proc. ACM SIGMOD,* vol. 22, no. 2, pp. 207-216, 1993.

[13] C. Kruegel, D. Mutz, W. Robertson, and F. Valeur, "Bayesian Event Classification for Intrusion Detection," *Proc. 19th Ann. Computer Security Applications Conf. (ACSAC 03),* pp. 14–23, 2003.

[14] K. Hwang, M. Cai, Y. Chen and M. Qin, "Hybrid Intrusion Detection with Weighted Signature Generation over Anomalous Internet Episodes," *IEEE Transactions on Dependable and Secure Computing,* vol. 4, no. 7, pp. 1–15.

[15] S. Kumar, *Classification and detection of computer intrusions.* Ph.D. thesis, Purdue Univ., West Lafayette, IN, 1995.

[16] W. Lee and D. Xiang, "Information-theoretic measures for anomaly detection," *Proc. 2001 IEEE Symposium on Security and Privacy,* pp. 130–143, 2001.

[17] D.S. Kim and J.S. Park, "Network-Based Intrusion Detection with Support Vector Machines, *Proc. Information Networking, Networking Technologies for Enhanced Internet Services International Conference (ICOIN 03)* pp. 747–756, 2003.

[18] K.K. Gupta, B. Nath and R. Kotagiri, "Layered Approach Using Conditional Random Fields for Intrusion Detection," *IEEE Transactions on Dependable and Secure Computing,* vol. 7, no. 1, pp. 35–49, 2010.

[19] C.J. Lin, S.Y. Hong, and C.Y. Lee, "Using Least Squares Support Vector Machines for Adaptive Communication Channel Equalization," *International Journal of Applied Science and Engineering,* vol. 3, no. 1, 51-59, 2005.

[20] X. Rui-Rui, B.G. Xing, G. Chen-Feng, and C.T. Lun, "Discussion about Nonlinear Time Series Prediction Using Least Squares Support Vector Machine," *Commun. Theor. Phys.,* vol. 43, no. 6, pp. 1056–1060, 2005.

[21] F. Wu, and Y. Zhao, "Least Square Support Vector Machine on Morlet Wavelet Kernal Function and its Application to Nonlinear System Identification," *Information Technology Journal,* vol. 5, no. 3, pp. 439–444, 2006.

[22] C. Quanhua, L. Zunxiong, and D. Guoqiang, "Facial Gender Classification with Eigenfaces and Least Squares Support Vector Machine," *Journal of Artificial Intelligence,* vol. 1, no. 1, pp. 28-33, 2008.

[23] X.D. Hoang, J. Hu, and P. Bertok, "A program based anomaly intrusion detection scheme using multiple detection engines and fuzzy inference, *Journal of Network and Computer Applications,* vol.32, Issue 6, pp. 1219-1228, 2009.

[24] F. Liu, Y. He, and L. Wang, "Application of Least Squares Support Vector Machine for Measurement of Soluble Solids Content of Rice Vinegars Using Vis/NIR Spectroscopy," *Proc. International Conference on Computational Intelligence and Security,* pp. 1044–1047, 2007.

[25] J. A. K. Suykens, and J. Vandewalle, "Multiclass least squares support vector machines," *Proc. International Joint Conference on Neural Networks,* pp. 900–903, 1999.

[26] J.A.K. Suykens, T.V. Gestel, J.D. Brabanter, B.D. Moor, and J. Vandewalle, *Least Square Support Vector Machine,* World Scientific, Singapore, 2002.

[27] J. Hu, D. Qiu, H.H. Chen, and X. Yu, “A simple and efficient data processing scheme for HMM based anomaly intrusion detection. *Special Issue of Advances on Network Intrusion Detection. IEEE Network,* vol. 23, no. 1, pp.42–47, 2009.

[28] J. Lafferty, A. McCallum, and F. Pereira, “Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data,” *Proc. 18th Intl Conf. Machine Learning (ICML 01),* pp. 282-289, 2001.

[29] C. Warrender, S. Forrest, and B. Pearlmutter, “Detecting Intrusions Using System Calls: Alternative Data Models,” *Proc. IEEE Symp. Security and Privacy (SP 99),* pp. 133-145, 1999.

[30] X. Chen, F. Zhang, W. Susilo, H. Tian, J. Li, and K. Kim, ”Identity-based chameleon hashing and signatures without key exposure.” *Information Sciences,* vol. 265, no. 1, pp. 198–210, 2014.

[31] X. Huang, Y. Xiang, E. Bertino, J. Zhou, and L. Xu, ”Robust Multi-Factor Authentication for Fragile Communications. *IEEE Transactions on Dependable and Secure Computing,* vol 11, no. 6, pp. 568–581, 2014.

[32] D. Wang, Z. Zhang, P. Wang, J. Yan, and X. Huang, ”Targeted Online Password Guessing: An Underestimated Threat,” *ACM Conference on Computer and Communications Security,* pp. 1242-1254, 2016

[33] J. Huang, M. Peng, H. Wang, J. Cao, G. Wang, X. Zhang, ”A Probabilistic Method for Emerging Topic Tracking in Microblog Stream,” *World Wide Web*, doi:10.1007/s11280-016-0390-4, pp. 1-26, 2016

[34] M.E. Kabir, H. Wang, and E. Bertino, ”Efficient systematic clustering method for k-anonymization,” *Acta Informatica*, vo. 48, no. 1, pp. 51-66, 2011.

[35] H Wang, J Cao, and Y Zhang, ”A flexible payment scheme and its role-based access control”, *IEEE Transactions on knowledge and Data Engineering*, vo. 17, no. 3, 425–436, 2005.

[36] H. Debar, M. Becke, and D. Siboni, “A Neural Network Component for an Intrusion Detection System, *Proc. IEEE Symp. Research in Security and Privacy (RSP 92),* pp. 240-250, 1992.

[37] Y. Bouzida and S. Gombault, “Eigenconnections to Intrusion Detection,” *Security and Protection in Information Processing Systems,* pp. 241-258, 2004.

[38] Z. Zhang, J. Li, C.N. Manikopoulos, J. Jorgenson, and J. Ucles, “HIDE: A Hierarchical Network Intrusion Detection System Using Statistical Preprocessing and Neural Network Classification,” *Proc. IEEE Workshop Information Assurance and Security (IAW 01),* pp. 85-90, 2001.

[39] E. Kabir, A. Mahmood, H. Wang, A. Mustafa, “Microaggregation Sorting Framework for K-Anonymity Statistical Disclosure Control in Cloud Computing,” *IEEE Transactions on Cloud Computing,* vol. PP, no.99, pp.1-1.

[40] Y. Zhang, Y. Shen, H. Wang, J. Yong, X. Jiang, “On Secure Wireless Communications for IoT under Eavesdropper Collusion,” *EEE Transactions on Automation Science and Engineering,* Vol. 13, pp: 1281-1293, 2016.

[41] J. Ma, L. Sun, H. Wang, Y. Zhang and U. Aickelin, “Supervised Anomaly Detection in Uncertain Sensor Data Streams,” *ACM Transactions on Internet Technology (TOIT),* 16, 1, Article 4 (January 2016), 20 pages. DOI= http://dx.doi.org/10.1145/2806890

[42] J. Zhang, H. Li, X. Liu, Y. Luo, F. Chen, H. Wang, L. Chang, “On Efficient and Robust Anonymization for Privacy Protection on Massive Streaming Categorical Information,” *IEEE Transactions on Dependable and Secure Computing*, no.1, pp. 1, 01/2015.

[43] Y. Zhang, Y. Shen, H. Wang, Y. Zhang, X. Jiang, “On Secure Wireless Communications for Service Oriented Computing,” *IEEE Transactions on Services Computing,* no. 1, pp. 1.

[44] S. Siuly, and Y. Li, Y., ”A novel statistical algorithm for multiclass EEG signal classification”, *Engineering Applications of Artificial Intelligence*, vol. 34, pp.154–167, 2014.

[45] S. Siuly, Y. Li, and P. Wen, "Identification of motor imagery tasks through CC-LR algorithm in brain computer interface", *International Journal of Bioinformatics Research and Applications*, vol. 9, no. 2, pp.156–172, 2013.

[46] M. Sabhnani and G. Serpen, "Application of Machine Learning Algorithms to KDD Intrusion Detection Dataset within Misuse Detection Context," *Proc. Intl Conf. Machine Learning, Models, Technologies and Applications (MLMTA 03),* pp. 209-215, 2003.

[47] *Results of the KDD99 Classifier Learning Contest,* http://www.cse.ucsd.edu/users/elkan/clresults.html, 2012.

[48] Y. Liu, K. Chen, X. Liao, and W. Zhang: "A Genetic Clustering Method for Intrusion Detection", Pattern Recognition, Vol. 37, Issue 5, pp. 927-942. 2004.

[49] H.G. Kayacik, A.N. Zincir-Heywood, and M.I. Heywood: "On the capability of an SOM based intrusion detection system", *Proceedings of the International Joint Conference on Neural Networks,* Vol. 3, pp. 1808–1813, 2003.

[50] T. Ambwani, "Multi class support vector machine implementation to intrusion detection", *Proceedings of the International Joint Conference on Neural Networks,* Vol. 3, pp. 2300-2305, 2003.

[51] V. Vapnik, *The Nature of Statistical Learning Theory,* Springer-Verlag, New York, 1995.

[52] *KDD Cup 1999 Intrusion Detection Data,* http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, 2012.

[53] *LS-SVMlab Toolbox (Version 1.8),* http://www.esat.kuleuven.ac.be/sista/lssvmlab/, 2012