Intelligent Decision Technologies 5 (2011) 347–356
DOI 10.3233/IDT-2011-0117
IOS Press

Network intrusion detection system: A machine learning approach

Mrutyunjaya Panda[a,*], Ajith Abraham[b], Swagatam Das[c] and Manas Ranjan Patra[d]
[a]*Department of EEE GITA, Bhubaneswar, Odisha, India*
[b]*MIR Labs, Washington, USA*
[c]*Department of ECE, Jadavpur University, Kolkata, India*
[d]*Department of Comp. Sc., Berhampur University, Odisha, India*

Abstract. Intrusion detection systems (IDSs) are currently drawing a great amount of interest as a key part of system defence. IDSs collect network traffic information from some point on the network or computer system and then use this information to secure the network. Recently, machine learning methodologies are playing an important role in detecting network intrusions (or attacks), which further helps the network administrator to take precautionary measures for preventing intrusions. In this paper, we propose to use ten machine learning approaches that include Decision Tree (J48), Bayesian Belief Network, Hybrid Naïve Bayes with Decision Tree, Rotation Forest, Hybrid J48 with Lazy Locally weighted learning, Discriminative multinomial Naïve Bayes, Combining random Forest with Naïve Bayes and finally ensemble of classifiers using J48 and NB with AdaBoost (AB) to detect network intrusions efficiently. We use NSL-KDD dataset, a variant of widely used KDDCup 1999 intrusion detection benchmark dataset, for evaluating our proposed machine learning approaches for network intrusion detection. Finally, Experimental results with 5-class classification are demonstrated that include: Detection rate, false positive rate, and average cost for misclassification. These are used to aid a better understanding for the researchers in the domain of network intrusion detection.

Keywords: Intrusion detection, machine learning, cost matrix

1. Introduction

Intrusion detection is defined as the process of monitoring the events occurring in a computer system or network and analyzing them for signs of intrusions [32]. The need for effective intrusion detection mechanism for computer systems was recommended by Denning and Neumann [5] in order to find reasons for intrusion detection within a secure computing framework. The first major work in the area of intrusion detection was discussed by Anderson in [16] with an insight to the fact that certain types of intrusions to the computer system security could be identified through a detailed analysis of information contained in the system's audit trial. Three threats were identified by Anderson which could be: External Penetrations – as unauthorized users of the system, internal penetrations – as authorized system users who use the system in an unauthorized manner, and finally Misfeasors – an authorized user who try to exploit their access privileges. But, it is Denning [6], who proposed an intrusion detection model which is considered to be the fundamental core of most intrusion detection research in use today.

Approaches for intrusion detection can be broadly divided into two types: misuse detection and anomaly detection. In misuse detection system, all known types of attacks (intrusions) can be detected by looking into the predefined intrusion patterns in system audit traffic. In case of anomaly detection, the system first learns a normal activity profile and then flags all system events that do not match with the already established profile. The main advantage of the misuse detection is its capability for high detection rate with a difficulty in finding the new or unforeseen attacks. The advantage of anomaly detection lies in the ability to identify the

*Corresponding author. E-mail: mrutyunjaya@ieee.org.

ISSN 1872-4981/11/$27.50

novel (or unforeseen) attacks at the expense of high false positive rate.

In fact, intrusion detection is considered as a classification problem, namely, to identify the behaviour of the network traffic to fall either in normal or any one out of the four attack categories (for example, Probing, Denial of Service, User to Root, and Root to Local). Hence, the main motivation is to develop accurate classifiers that can effectively classify the intrusive behaviour than the normal one.

Despite number of approaches based on different soft computing paradigms have been proposed for intrusion detection, the possibilities of using the techniques are still considered to be under utilized.

In this paper, we propose to use NSL-KDD dataset [30] in place of most widely used KDDCup 1999 benchmark intrusion detection dataset [1] for building a network intrusion detection system. It is also pointed out that the NSL-KDD dataset contains selected records of the complete KDDCup 1999 dataset and does not suffer from the any of the inherent problems. In order to conduct a thorough analysis of the proposed research, we apply ten machine learning approaches with various evaluation methods, including cost sensitive classification, to build a network intrusion detection system.

The reminder of this paper is organized as follows. Section 2 provides the review of the related research in the filed of intrusion detection. The proposed methodologies along with the performance evaluation measures are presented in Section 3. In Section 4, we highlight about the NSL-KDD dataset with a discussion about the inherent problems of most commonly used KDDCup 1999 dataset followed by the Experimental results and discussion in Section 5. Finally, Section 6 concludes the paper unfolding a few future directions of research.

2. Related research

Machine learning methodologies are being widely used by the researchers in the field of network intrusion detection due to their generalization capabilities that helps to understand the technical knowledge about the intrusions that do not have any predefined patterns. Earlier studies have utilized association rules to detect network intrusions [7,29,41,42]. The drawback of using association rule mining is that they tend to produce a large number of rules after using many performance measures to the whole rule set obtained and thereby, increase the complexity of the system. Recently, the researchers shifted their focus on learning using examples and machine learning techniques to build an efficient classifier. Neural networks have been extensively used to identify both anomaly and misuse patterns in [9,35]. The difficulty in neural networks is that they need to be trained for a longer period of time while applied in a large dataset like KDDCup 1999. Decision trees [12,31] and Naïve Bayes [23] are considered to be among the well known machine learning methodologies used in IDS. Decision Trees are used because of its simplicity, fast adaptability and most importantly to produce high classification accuracy. At the same time, Naïve Bayes assumes the conditional independence of data feature, instead of correlated features which may degrade its performance. Nguyen and Choi [10] proposed a classification model by applying top ten classifier algorithms to network intrusion detection. Mill and Inoue [15] use successful support vector machine (SVM) learning approach to classify network requests, where, the authors failed to perform the 5-class classification to get an insight about the detailed accuracy of each class in IDS that either fall in normal or any one of the attack category, which is highly required for the system administrator to obtain knowledge and take further actions in order to prevent intrusions in future. In [36], Abraham et al. investigated some new techniques by exploring the ensemble and hybrid approach using Decision tree (DT) and Support Vector Machines (SVM) for intrusion detection and concluded that the hybrid approach improves the performance for all classes in comparison to a direct SVM technique. Zhang et al. [13] apply the random forests algorithm in misuse, anomaly and hybrid detection, where they found that their proposed hybrid framework achieve better detection rate with low false positive rate than the others. Ramos and Abraham [39] presents a self-organized and Ant Colony Based Intrusion Detection System (ANTIDS) to detect network intrusions and compare their findings with the other well known algorithms like Decision trees, SVM and linear genetic programming for an efficient NIDS. Panda and Patra [24] explored the hybrid clustering approach by combining conceptual clustering (COBWEB) with Farthest First Traversal (FFT) clustering to enhance the detection rate of the attacks that fall under rare attack categories. A new approach to intrusion detection using artificial neural network and fuzzy clustering with a meta-learner is proposed in [8], which outperforms back propagation neural network and others in terms of precision and stability. Recently, Tavallaee et al. [25] reviewed the current state of experimental practice in the area of IDS

and provide a summary of their observation by highlighting the common pitfalls among the survey of 267 studies.

All the above researches use KDDCup 1999 dataset for their experimentation. However, the authors in [14, 26,27] provided a detailed analysis of the KDDCup 1999 dataset and focussed on their shortcomings. In order to solve those issues, the authors in [27] proposed a new dataset, NSL-KDD, which does not suffer from any of the shortcoming as mentioned. The authors use Discriminative Multinomial Naïve Bayes (DMNB) using NSL-KDD dataset to build an efficient network intrusion detection system [28].

Following this line of thinking, we propose to apply ten significant machine learning algorithms, using NSL-KDD dataset with separate training and testing set to evaluate their performances in detecting network intrusions.

3. Proposed methodologies and evaluation

In this section, we elaborate our machine learning approaches using NSL-KDD dataset to build a NIDS with various evaluation measures in order to find the efficacy of the model.

3.1. Machine learning methodologies

Naïve Bayes (NB) is called as Idiot's Bayes, Simple Bayes and Independent Bayes, which is popular for its simplicity, elegance and robustness in building a classifier. Naïve Bayes takes a small amount of training data to estimate the means and variances of each class under consideration rather than the entire covariance matrix [18]. Further, the assumption of considering all attributes conditionally, independent for a given class provides some relaxation to improve its classification performance.

Decision trees (DT) is one of the most commonly used machine learning approach in the field of intrusion detection, which consists of decision and leaf nodes. Decision node represents to testing a single attribute of the given instances whereas the leaf node presents the idea about whether the output of a classifier falls in to either normal or intrusion (any of the possible attacks) category during the classification phase. The essence of using the DT lies in its ability to handle large dataset effectively and to detect the new or unforeseen attacks. We use J48, a variant of earlier version of C4.5 proposed by J. Ross Quinlan for our experimentation [4].

Support Vector Machines (SVM) is based on intuitive geometric principles that linearly separates the training data to obtain the minimum expected risk for the application, as per Vapnik in 1995 [40]. As all problems cannot be separated linearly, SVM use some kernel functions like polynomial or radial basis function in order to transform the linear algorithms to nonlinear ones via a map into feature spaces. In this paper, we use polynomial kernel to obtain the support vectors along the surface of this function which further can be used by SVM for necessary classification on data.

Random Forests [21] are an ensemble of unpruned classification trees, where each tree is constructed using separate bootstrapped sample from the original data. This way random forest generates many classification trees. To classify a new object from an input vector, we have to put the input vector down each of the tree in the forest. Here, each tree gives a vote for a class type and then the forest chooses the class that are having the most votes. In random forest, there is no need for cross validation or separate test set to get an error estimate. It is estimated internally, during the simulation as each tree is constructed using sampling with replacement, where about one third of the total instances are left out of the sample and not used in the construction of a particular tree. These samples or data are called as oob (out-of-bag) data. This oob data can further be used to get an unbiased estimate of classification error as trees are added to the forest. In this paper, we use random forest oob (out-of-bag) error estimate for its effectiveness in learning from the whole training dataset provided. The importance of this method lies in handling the large dataset efficiently and therefore considered one of the most accurate among current machine learning algorithms.

Rotation Forest [34] is a new classifier ensemble method which is used to encourage simultaneously individual accuracy and diversity within the ensemble. Accuracy is sought by considering the Naïve Bayes in place of Principal component analysis (PCA) as a supervised filter applied to each subset which is applied to whole dataset to train each base classifier. In contrast, each base classifier (we use J48, DT) obtains diversity through feature extraction.

It is observed that different classifiers provide complimentary information about the patterns to be classified. Even, one classifier can outperform the other in a particular problem domain. Hence, the idea of ensemble of classifiers came into picture to enhance the overall classification accuracy. The AdaBoost algorithm proposed by Freund and Schapire [43] is one of the

most widely used ensemble methods for its simplicity, very accurate prediction. However, the success lies in the accuracy and diversity of the base classifiers used.

Dagging [19] creates a number of disjoint, stratified folds from the whole dataset and feeds each data to a copy of the used base learners, in which majority voting is used for the proposed classification problems. In this, we use Dagging with support vector machine and polynomial kernel for our experiments.

Bayesian Belief Networks (BBN) is a popular method for dealing with uncertainty applied successfully in many real world applications. BBN is a directed acyclic graph and its structural representation is represented by nodes that correspond to random variables in a problem domain. Nodes in a BBN contain states of random variables [20]. In this paper, we use BBN with Tabu search as a feature selection algorithm to perform the classification to detect network intrusions efficiently.

Further, we use discriminative multinomial Naïve Bayes (DMNB) with supervised data filtering by nominal to binary process. The DMNB uses a simple, efficient and effective discriminative parameter learning method that learns parameters by discriminatively computing frequencies from intrusion data. It is found computationally efficient, converges quickly and more importantly does not overfit which makes this method a promising one for our proposed method.

Finally, we tried to perform Locally Weighted Learning (LWL) with Linear Nearest Neighbour (LNN) search with Decision Stumps (DS) search as a filtering scheme with J48 DT to detect network intrusions. Our ultimate goal is to improve the accuracy of the built model. In local learning, each local model is trained entirely independently of all other local models such that it does not affect directly to compute the complexity of the model. This property avoids overfitting if a robust learning scheme like LNN exists for training the individual local model. Further, LWL is able to deal with a large number of inputs that are possibly redundant or irrelevant [37].

3.2. Evaluation measures

In this section, we provide a detailed evaluation of the machine learning techniques with various performance measures to detect network intrusions.

- **Detection Rate and Recall Rate** are two widely used measures for evaluating the quality of results in statistical classification domain [38]. Detection rate (Precision Rate) is defined as a measure of fidelity or exactness which provides an insight to understand how much efficiently all class labels are detected truly. In other words, Detection Rate (DR) is calculated as the ratio between the true positives to the sum of true positives and false positives. True positives are the number of items correctly classified as belonging to the class and which are incorrectly classified are termed as false positives. In contrast, recall (RR) is a measure of completeness. RR can be defined as the number of true positives divided by the total number of items that actually belong to the class.
- **False Positive Rate** (**FPR**) **and False Negative Rate** (**FNR**) are also important in finding the performance of any machine learning algorithms. False positive (FP) occurs when actually there is no attack occurred but alarm is raised, where as false negative (FN) gives an indication of a possible attack but there is no alarm raised.
- **Cost sensitive classification** is also considered as an important performance evaluation measures in order to find the cost of misclassification for intrusion detection. In this, with a given test set, the average cost of a classifier is calculated as follows:

$$Cost = \frac{\sum_{i=1}^{5}\sum_{j=1}^{5} ConfM(i,j)^{*} CostM(i,j)}{N},$$

Where, *ConfM*(i, j) denotes the entry at row i, column j in the confusion matrix, *CostM*(i, j) is the entry at row i, column j in the cost matrix and N is the total number of connections in the intrusion detection test dataset. Assessments on numbering of network attacks categories along with the cost matrix are shown below in Tables 1 and 2 respectively [3].

Table 1
Numbering of the attack categories [3]

0	Normal
1	Probe
2	DoS
3	U2R
4	R2L

Table 2
Cost matrix [3]

	Normal	Probe	DoS	U2R	R2L
Normal	0	1	2	2	2
Probe	1	0	2	2	2
DoS	2	1	0	2	2
U2R	3	2	2	0	2
R2L	4	2	2	2	0

Table 3
Confusion matrix (Actual predictions)

	Predicted class positive (Normal)	Predicted class negative (Attack)
Actual Class Positive (Normal)	A(TN)	B(FP)
Actual Class Negative (Attack)	C(FN)	D(TP)

Table 4
Confusion matrix (Expected predictions)

	Predicted class positive	Predicted class negative
Actual Class Positive	E	F
Actual Class Negative	G	H

- **Run Time** is the time taken by the classifier to build a model during its training phase.
- **Bias-Variance Dilemma** is another measure that plays a vital role in the evaluation of a classifier. The principle can be stated as: datasets with too few parameters are inaccurate because of a large bias and thus don't have enough flexibility, where as large variance occurs when there are too many parameters that indicate too sensitivity to the sample.
- **F-Measure, Kappa and Root mean square error** (**RMSE**) are also among other popular measures to evaluate the performance of a classifier. The Kappa statistic is used to measure the agreement between predicted and observed categorization of the used dataset, while correcting for arguments that occur by chance. Kappa statistic takes the expected figure into account, then subtracts it from the predictor's success and finally, expresses the result as a proportion of the total for a perfect predictor, as proposed by Witten and Frank [11]. The Kappa statistic is calculated as:

$$Kappa = \frac{(A+D)-(E+H)}{(A+B+C+D)-(E-H)} * 100\%$$

Where, the values of A to H can be obtained from the confusion matrix of actual and expected predictions given in Tables 3 and 4 respectively.

Kubat and Matwin [22] points out that accuracy alone cannot be considered as sole reliable measure for classification. This is because in a case where there are 15 instances, out of which 13 are negative and 2 are positive, if the classifier identifies all of them as negative, then accuracy will be 87%. However, it would result in ignoring all the positive instances. Therefore, the F-measure is calculated as the harmonic mean of precision and recall to address this problem, as:

$$F - measure = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision + Recall}}$$

4. Datasets used

4.1. KDDCup 1999 dataset

In 1998 and 1999, Lincoln Laboratory at Massachusetts Institute of Technology (MIT) engaged in a network intrusion detection project sponsored by Defence Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory (ARFL) [24,25]. The KDDCup 1999 dataset is a subset of DARPA dataset prepared by Stolfo and Lee [42], which can be directly used for testing and evaluation of the classifiers without further pre-processing. Each connection record in the KDDCup 1999 dataset is labeled as either normal or anomalous (attack). There are 39 types of attacks which are grouped into four major categories: DoS (Denial of Service), U2R (unauthorized access to root privileges), Probing, and R2L (unauthorized access from a remote machine).

4.2. NSL-KDD dataset

During last decade, KDDCup 1999 intrusion detection benchmark dataset is used by many researchers in order to build an efficient network intrusion detection system. However, recent study shows that there are some inherent problems present in KDDCup 1999 dataset [27]. The first important limitation in the KDDCup 1999 dataset is the huge number of redundant records in the sense that almost 78% training and 75% testing records are duplicated, which cause the learning algorithm to be biased towards the most frequent records, thus prevent it from recognizing rare attack records that fall under U2R and R2L categories. At the same time, it causes the evaluation results to be biased by the methods which have better detection rates on the frequent records. This new dataset, NSL-KDD provided in [27] is used for our experimentation and is now publicly available for research in intrusion detection. It is also stated that though the NSL-KDD dataset still

Table 5
Known and novel attack types [2]

Attack category	Probing attacks	DoS attacks	U2R attacks	R2L attacks
Known Attacks	ipsweep, satan, nmap, portsweep	Teardrop, pod, land, back, Neptune, smurf	Perl, loadmodule, rootkit, buffer_overflow	ftp_write, phf, guess_passwd, warezmaster, warezclient, imap, spy, multihop
Novel Attacks	saint, mscan	mailbomb, udpstorm, apache2, processtable	Xterm, ps, sqlattack, httptunnel	Named, snmpguess, worm, snmpgetattack, xsnoop, xlock, sendmail

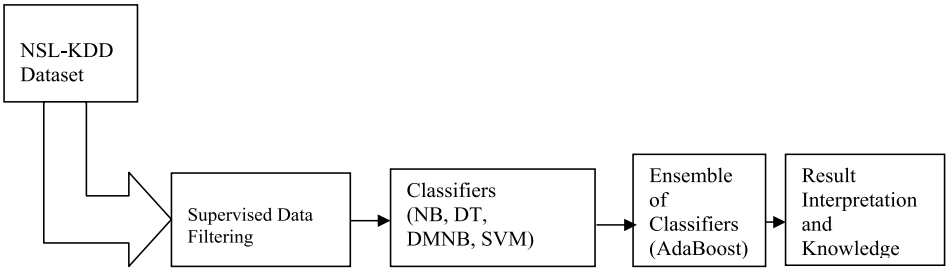

Fig. 1. The proposed framework of our Network Intrusion Detection System.

suffers from some of the problems discussed in [14] and may not be a perfect representative of existing real networks, it can be applied an effective benchmark dataset to detect network intrusions. More details about the inherent problems found in KDDCup dataset can be obtained from [27]. In this NSL-KDD dataset, the simulated attacks can fall in any one of the following four categories.

- Probing Attack: This is a type of attack which collect information of target system prior to initiating an attack. Some of the examples are Satan, ipsweep, nmap attacks.
- DoS Attack: Denial of Service (DoS) attack results by preventing legitimate requests to a network resource by consuming the bandwidth or by overloading computational resources. Examples of this are Smurf, Neptune, Teardrop attacks.
- User to Root (U2R) Attack: In this case, an attacker starts out with access to a normal user account on the system and is able to exploit the system vulnerabilities to gain root access to the system. Examples are eject, load module and Perl attacks.
- Root to Local (R2L) Attack: In this, an attacker who doesn't have an account on a remote machine sends packet to that machine over a network and exploits some vulnerabilities to gain local access as a user of that machine. Some examples are ftp_write, guess password and imap attacks.

The known and novel attack types supported in NSL-KDD dataset are presented in Table 5.

5. Experimental setup, results and discussion

In this section, the experimental results of our ten machine learning techniques with five class classification methodology using NSL-KDD intrusion detection dataset are provided in order to detect network intrusions and then comparison with the existing approaches is done to evaluate the efficacy of our network intrusion detection model. We use Weka [11] for our experimentation in a Pentium-4 Machine with 2.86GHz CPU, and 1GB RAM. The framework of the proposed approach is shown in Fig. 1.

5.1. Experimental results and discussions

All the experiments are conducted using NSL-KDD dataset that has 60438 training instances, 22544 instances for testing with 42 attributes and 38 attack types for five class classifications to build an efficient network intrusion detection system. We have evaluated all our algorithms with various evaluation measures, as discussed in Section 3.2 and the results are presented in Tables 6 to 11.

From Table 6, it can be observed that J48, NB + J48 and Lazy LWL + LNN + J48 provides highest recall rate of 68.8% in detecting normal activities; 89.6% for probing attacks in ensemble of J48 with AdaBoost; 98.4% for DoS attacks in combination of BBN, TS and SE; 100% for U2R attacks in case of both RotFor + NB + J48 and NB + Random Forest; 96.7% in NB + Random Forest for detecting R2L attacks.

Table 6
Recall rate comparison

Methodology/Recall Rate	Normal	Probe	DoS	U2R	R2L
NB + J48	0.688	0.748	0.277	0.692	0.958
AB + NB + J48	0.651	0.86	0.958	0.5	0.921
RotFor + NB + J48	0.657	0.721	0.972	1.0	0.918
BBN + TS + SE	0.672	0.675	0.984	0.47	0.257
Dagging + SMO + Poly Kernel	0.643	0.54	0.92	0.0	0.75
J48	0.688	0.749	0.959	0.75	0.952
AB + J48	0.665	0.896	0.961	0.615	0.76
Nom2Bin + DMNB	0.644	0.843	0.908	0.0	0.0
NB + Random Forest	0.679	0.849	0.935	1.0	0.967
Lazy LWL + LNN + J48	0.688	0.748	0.96	0.75	0.952

Table 7
F-Value comparison

Methodology/$F-$value	Normal	Probe	DoS	U2R	R2L
NB + J48	0.806	0.728	0.414	0.085	0.108
AB + NB + J48	0.78	0.78	0.852	0.038	0.026
RotFor + NB + J48	0.783	0.711	0.851	0.03	0.04
BBN + TS + SE	0.793	0.67	0.847	0.998	0.076
Dagging + SMO + Poly Kernel	0.761	0.536	0.828	0	4.3E-03
J48	0.806	0.728	0.883	0.085	0.11
AB + J48	0.79	0.775	0.889	0.075	0.013
Nom2Bin + DMNB	0.775	0.579	0.853	0	0
NB + Random Forest	0.799	0.737	0.882	0.04	0.062
Lazy LWL + LNN + J48	0.806	0.728	0.884	0.084	0.107

Table 8
Performance evaluation of various classifiers

Methodology	Kappa	RMSE	Build time in seconds	Number of leaves and trees	Average cost of misclassification
NB + J48	0.5921	0.1193	109.08	547, 662	0.643
AB + NB + J48	0.5868	0.119	777.92	697, 830	0.711
RotFor + NB + J48	0.583	0.1198	1350	867, 1027	0.7
BBN + TS + SE	0.572	0.1196	64.44	**********	0.698
Dagging + SMO + Poly Kernel	0.5379	0.155	859.56	**********	0.757
J48	0.5921	0.1193	98.44	547, 662	0.686
AB + J48	0.591	0.1186	980.22	1023,1142	0.679
Nom2Bin + DMNB	0.5443	0.1153	73.66	**********	0.742
NB + Random Forest	0.5904	0.1105	67.67	RF of 10 trees, with 6 random features and oob error = 0.03	0.671
Lazy LWL + LNN + J48	0.5921	0.1193	104.02	547, 662	0.643

From Table 7, it is evident that highest F-value is obtained in NB + J48, J48 and Lazy LWL + LNN + J48 with 80.6%; 78% in AB + NB + J48; 88.9% in AB + J48; BBN + TS + SE with 99.8%; J48 with 11% in detecting probing attacks, DoS attacks, U2R attacks and R2L attacks respectively.

Table 8 presents the performance evaluations of various classifiers with their corresponding high kappa value, low root mean square error, less time to build the model and more importantly low cost of misclassification. From Table 8, Lazy LWL + LNN + J48 and NB + J48 provides low cost for mis-classification with 0.643 where as Lazy LWL + LNN + J48 provides better performance with same kappa value of 59.21%, 11.93% of RMSE, 547 leaves and 662 trees but relatively faster than the combination of NB + J48.

False negative rate comparison amongst all the classifiers is provided in Table 9. Low false negative rate is obtained with 31.1% in case of LazyLWL + LNN + J48; 10.4% with AB + J48; 1.6% with BBN + TS + SE,; 0% with RotFor + NB + J48, Dagging + SMO + PolyKernel, Nom2Bin + DMNB and NB + Random Forest; 3.3% with NB + Random Forest for detecting Normal, Probing attack, DoS attack, U2R attack and R2L attacks respectively.

From Table 10, it can be seen that AB + J48 provides high detection rate of 97.4% for normal category, where as the highest detection rate for probing attacks, DoS

Table 9
False negative rate comparison

Methodology/false negative rate	Normal	Probe	DoS	U2R	R2L
NB + J48	0.312	0.251	0.039	0.307	0.042
AB + NB + J48	0.349	0.139	0.041	0.5	0.078
RotFor + NB + J48	0.343	0.278	0.083	0	0.082
BBN + TS + SE	0.327	0.324	0.016	0.53	0.742
Dagging + SMO + Poly Kernel	0.357	0.459	0.079	0	0.25
J48	0.312	0.251	0.04	0.25	0.048
AB + J48	0.335	0.104	0.038	0.384	0.24
Nom2Bin + DMNB	0.356	0.157	0.092	0	1
NB + Random Forest	0.321	0.151	0.064	0	0.033
Lazy LWL + LNN + J48	0.311	0.251	0.039	0.25	0.048

Table 10
Detection rate comparison

Methodology/detection rate	Normal	Probe	DoS	U2R	R2L
NB + J48	0.972	0.71	0.819	0.045	0.057
AB + NB + J48	0.973	0.714	0.767	0.02	0.013
RotFor + NB + J48	0.969	0.702	0.756	0.015	0.02
BBN + TS + SE	0.966	0.665	0.745	0.19	0.045
Dagging + SMO + PolyKernel	0.931	0.532	0.752	0	2.17E-03
J48	0.972	0.709	0.819	0.045	0.057
AB + J48	0.974	0.682	0.828	0.04	6.89E-03
Nom2Bin + DMNB	0.973	0.441	0.803	0	0
NB + Random Forest	0.973	0.652	0.835	0.02	3.20E-02
Lazy LWL + LNN + J48	0.972	0.709	0.819	0.045	0.057

Table 11
False positive comparison

Methodology/false positive rate(FPR)	Normal	Probe	DoS	U2R	R2L
NB + J48	0.056	0.034	0.078	8.4E-03	0.103
AB + NB + J48	0.083	0.033	0.096	8.6E-03	0.107
RotFor + NB + J48	0.084	0.035	0.098	8.66E-03	0.107
BBN + TS + SE	0.075	0.04	0.101	7.17E-03	0.108
Dagging + SMO + Poly Kernel	0.162	0.056	0.103	8.79E-03	0.108
J48	0.057	0.034	0.078	8.4E-03	0.1
AB + J48	0.066	0.036	0.074	8.45E-03	0.108
Nom2Bin + DMNB	0.089	0.06	0.087	8.79E-03	0.109
NB + Random Forest	0.059	0.039	0.074	8.62E-03	0.106
Lazy LWL + LNN + J48	0.057	0.034	0.078	8.4E-03	0.104

attacks, U2R attacks and R2L attacks are obtained from the combination of AB + NB + J48 with 71.4%, NB + Random Forest with 83.5%, BBN + TS + SE with 19% and Lazy LWL + LNN + J48, J48, NB + J48 with 5.7% respectively.

It can be observed from Table 11 that NB + J48 provides lowest FPR of 5.6% for detecting normal instances, 3.3% with AB + NB + J48 for Probing attacks, 7.4% with both NB + Random Forest & AB + J48 for DoS attacks, 7.17E-03 with BBN + TS + SE for U2R attacks and 10% with J48 for R2L attacks.

We have also provided the results the current research with KDDCup 1999 dataset and NSL-KDD dataset in Tables 12 and 13 respectively. Though, using KDDCup 1999 dataset, the algorithms provide better detection accuracy in comparison to our approaches using NSL-KDD dataset, it cannot be considered for industrial use [26], because of its inherent limitations. It is also said that NSL-KDD does not free from all problems as discussed in [14], still the results obtained from this can be relied for building an NIDS.

While comparing all the methodologies from the results provided, the best one is chosen based on the high detection rate, high recall rate, high F-value with low false positive rate, low false negative rate for normal as well as all attack categories. The results show that no single algorithm outperforms other in all respects. Further, the Low root mean square error (RMSE), high kappa value, less building time to construct a classifier, and moreover the low cost misclassification also plays

Table 12
Performance comparison with existing research using KDDCup 1999 dataset

	PNrule [33]		Columbia model [42]		MECF-STV [38]		SVM [17]		Hybrid DT + SVM [36]	
	DR %	FPR %	DR %	FPR %	DR %	FPR %	DR %	FPR %	DR %	FPR %
Normal	99.5	27.0	–	–	99.8	3.6	99.64	–	99.7	–
DoS	96.9	0.05	24.3		98.1	0.06	99.92		99.92	
Probe	73.2	7.5	96.7		99.3	1.1	98.57		98.57	
U2R	6.6	89.5	81.8		89.7	0.03	40.0		48.0	
R2L	10.7	12.0	5.9		48.2	0.19	33.92		37.8	

Table 13
Comparison with data mining methods using NSL-KDD dataset

Classifier	Detection accuracy (%)	Time taken to build the model in seconds	False alarm rate in %
Decision Trees (J48) [7]	81.05	****	****
Naïve Bayes [7]	76.56	****	****
Random forest [7]	80.67	*****	****
SVM [7]	69.52	*****	****
Discriminative Multinomial Naïve Bayes + N2B [28]	**96.5**	**1.11**	**3.0**

a significant role while selecting a suitable classifier for the designing an efficient network intrusion detection systems.

6. Conclusion and future scope

The empirical analysis from this research suggests that our proposed approach using various machine learning methodologies using NSL-KDD dataset performs reasonably well in all categories of majority attacks. However, a poor detection rate in detecting U2R and R2L minority attacks are inevitable because of their large bias available in the dataset. The low false positive rate obtained in classifying attacks as well as normal instances makes our approach more interesting. We also perform a cost sensitive classification and found the models achieve low misclassification cost. Finally, it can be noted that no system is absolutely secure with a given set of best possible algorithms, while protecting our resources from network attacks. This makes the computer security is always an active and challenging area of research. To move further in this direction, we propose to evaluate more machine learning algorithms to detect minority attacks efficiently with acceptable false appositive rate and low cost of misclassification in future.

References

[1] A.A. Ghorbani, W. Lu and M. Tavallaee, Network intrusion detection and prevention: Concepts and Techniques, *Advances in Information Security*, Springer, 2010.

[2] A.O. Adetunmbi, S.O. Falaki, O.S. Adewale and B.K. Alese, Network Intrusion Detection based on rough set and k-nearest neighbour, *Intl Journal of computing and ICT research* **2**(1) (2008), 60–66.

[3] C. Elkan, Results of the KDD'99 classifier learning, *SIGKDD Explorations* **1**(2) (2000), 63–64.

[4] C. Krugel and T. Toth, Using decision tree to improve signature based intrusion detection, in: *Proceedings of RAID*, (Vol. 2820), G. Vigna, E. Jonsson and C. Kruegel, eds, Lecture Notes in Computer Science, 2003, pp. 173–191.

[5] D.E. Denning and P.G. Neumann, Audit trail analysis and usage data collection and processing, Technical report project 5910, SRI International, 1985.

[6] D.E. Denning, An intrusion detection model, *IEEE Trans. On Software Engineering* **SE-13**(2) (1987), 118–131. IEEE Computer Society Press, USA.

[7] D. Barbara, J. Couto, S. Jajodia, L. Popyack and N. Wu, ADAM: Detecting intrusions by data mining, in: *Proceedings of 2nd Annual IEEE workshop on Infor Assu Secur*, New York, Jun 2001, pp. 11–16.

[8] G. Wang, J. Hao, J. Ma and L. Huang, A new approach to intrusion detection using artificial neural networks and fuzzy clustering, *Expert system with applications* **37** (2010), 6225–6232, Elsevier.

[9] H. Debar and B. Dorizzi, A neural network component for an intrusion detection system, in: *Proceedings of the IEEE Computer Society. Symposium on research in security and privacy,* Oakland, CA, May 1992, pp. 240–250.

[10] H.A. Nguyen and D. Choi, Application of data mining to network intrusion detection: classifier selection model, in: *Proceedings of Challenges for Next Generation Network Operations and Service Management* (*APNMOS 2008*), (Vol. 5297), Y. Ma, D. Choi and S. Ata, eds, Lecture notes in computer science, Springer, 2008, pp. 399–408.

[11] H. Ian, Witten and Eibe Frank, *Data Mining-Practical Machine Learning Tools and Techniques*, (2nd ed.), Elsevier, 2005.

[12] J.-H. Lee, J.-H. Lee, S.-G. Sohn, J.-H. Ryu and T.-M. Chung, Effective value of decision tree with KDD 99 intrusion detection datasets for intrusion detection system, in: *Proceedings*

of 10th Intl. Conf. on Advanced Communication Technology (*ICACT 2008*), Feb 2008, pp. 1170–1175.

[13] J. Zhang, M. Zulkernine and A. Haque, Random Forests based network intrusion detection systems, *IEEE Trans on System, Man, Cybernetics, Part-C: Applications and Reviews* **38**(3) (2008), 649–659.

[14] J. McHugh, Testing Intrusion detection system: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory, *ACM Transaction on Information and system security* **3**(4) (2000), 262–294.

[15] John Mill and A. Inoue, Support vector classifiers and network intrusion detection, in: *Proceedings of 2009 IEEE Intl. Conf. on Fuzzy System*, WA, USA, 1, 2004, pp. 407–410.

[16] J.P. Anderson, *Computer Security Threat Monitoring and Surveillance*, Technical report, Contract 79F296400, J.P. Anderson company, Fort Washington, Pennsylvania, 1985.

[17] J. Su, H. Zhang, C.X. Ling and S. Matwin, Discriminative parameter learning for Bayesian networks, in: *ACM Proceedings of 25th International Conference on Machine Learning* (*ICML 2008*), H. Finland, W.W. Cohen, A. McCallum and S.T. Roweis, eds, 307, Jun 2008, pp. 1016–1023.

[18] K. Huang, I. King and M.R. Lyu, Finite mixture model of bounded semi-naïve Bayesian networks classifier, in: *Proceedings of 10th Intl. Conference on Neural Information Processing* (*ICONIP-2003*), Istanbul, Turkey, Jun 2003, Okyay Kaynak, Ethem Alpaydin, Erkki Oja, Lei Xu eds, Lecture notes in computer science, 2714, pp. 115–122.

[19] K.M. Ting and I.H. Witten, Stacking bagged and dagged models, in: *Proceedings of 14th Intl Conf on Machine Learning*, San Francisco, CA, 1997, pp. 367–375.

[20] K.-C. Khor, C.-Y. Ting and S.-P. Amnuaisuk, From feature selection to building of Bayesian classifiers: A network intrusion detection perspective, *American Journal of applied sciences* **6**(11) (2009), 1949–1960.

[21] L. Breiman, Random Forests, *Machine Learning* **45** (2001), 5–32.

[22] M. Kubat and S. Matwin, Addressing the curse of imbalanced training sets: one sided selection, in: *Proceedings of the 14th International conference on Machine Learning*, D.H. Fisher, ed., Nashville, Tennessee, USA, July 1997, pp. 179–186. Morgan Kauffman Publisher.

[23] M. Panda and M.R. Patra, Network intrusion detection using Naïve Bayes, *International Journal of Computer Science and Network Security* **7**(12) (2007), 258–263.

[24] M. Panda and M.R. Patra, A Hybrid Clustering approach for network intrusion detection using COBWEB and FFT, *Journal of Intelligent System* **18**(3) (2009), 229–245.

[25] M. Tavallaee, N. Stakhanova and A.A. Ghorbani, Toward credible evaluation of anomaly based intrusion detection methods, *IEEE Trans On Syst, Man, and Cybernetics, Part-c: Applications and Reviews* **40**(5) (2010), 516–524.

[26] M. Mahoney and P. Chan, An analysis of the 1999 DARPA/Lincoln Lab. Evaluation data for network anomaly detection, in: *Proceedings of 6th International Symposium Recent Advances in Intrusion Detection* (*RAID-03*), (Vol. 2820), Pittsburgh, USA, lecture notes in computer science, Sept 2003, pp. 220–238. Springer.

[27] M. Tavallaee, E. Bagheri, W. Lu and A.A. Ghorbani, A detailed analysis of the KDD Cup 1999 dataset, in: *Proc. of 2009 IEEE International Symposium on Computational Intelligence in Security and Defence Applications* (*CISDA-2009*), Ottawa, ON, July 2009, 1–6, IEEE Press.

[28] M. Panda, A. Abraham and M.R.Patra, Discriminative multinomial Naïve Bayes for network intrusion detection, in: *Proceedings of 6th Intl. Conf. on Information Assurance and Security* (*IAS-2010*), Aug. 2010, USA, 5–10, IEEE Press.

[29] M. Panda and M.R. Patra, Mining association rules for constructing network intrusion detection model, *International Journal of Applied Engineering and Research* **4**(3) (2009), 381–398.

[30] NSL KDD dataset, http://iscx.ca/NSL-KDD/, accessed on 8-10-2010.

[31] N.B. Amor, S. Benferhat and Z. Elouedi, Naïve Bayes Vs Decision Trees in Intrusion detection system, in: *Proceedings of ACM Symposium on Applied Computing*, 2004, pp. 420–424.

[32] R. Power, CSI/FBI computer crime and security survey, *Computer Security Journal* **18**(2) (2002), 7–30.

[33] R. Agarwal and M.V. Joshi, PNrule: A new framework for learning classifier models in data mining. *Technical Report*, Department of Computer Science, University of Minnesota, Report No-RC 21719, 2000.

[34] J.J. Rodriguez, L.I. Kuncheva and C.J. Alonso, Rotation Forest: A new classifier ensemble method, *IEEE Transaction on Pattern Analysis and Machine Intelligence* **28**(10) (2006), 1619–1630, IEEE Computer Society Press, USA.

[35] S. Mukkamala, G. Janoski and A.H. Sung, Intrusion detection using Neural network and support vector machines, in: *Proceedings of IEEE International Conf. on Neural Networks*, 2002, pp. 1702–1707, IEEE Computer Society Press, USA.

[36] S. Peddabachigari, A. Abraham, C. Grosan and J. Thomas, Modelling intrusion detection system using hybrid intelligent systems, *Journal of Network and Computer Applications* **30** (2007), 114–132.

[37] S. Vijaya Kumar and S. Schaal, Local dimensional reduction for locally weighted learning, in: *Proc. of IEEE Intl. Sympo. On Computational Intelligence in Robotics and Automation* (*CIRA-97*), Monterey, California, 220–225, IEEE Press.

[38] T.P. Tran, P. Tsai, T. Jan and X. Kong, *Network Intrusion Detection Using Machine Learning and Voting Techniques*, Y. Zhang, ed., 2010, In Tech Publisher.

[39] V. Ramos and A. Abraham, ANTIDS: self organised Ant-based clustering model for intrusion detection system, in: *Proceedings of 4th IEEE International Workshop on Soft Computing as Transdisciplinary Science and Technology* (*WSTST 2005*), Japan, Ajith Abraham et al., eds, Advances in Soft Computing, 29 (2005), 977–986.

[40] V.N. Vapnik, The nature of statistical learning theory, *Information and Statistics Series*, 1995, Springer.

[41] W. Lee and J. Stolfo, Data mining approach for intrusion detection, in: *Proceedings of the 7th USENIX security symposium* (*SECURITY-98*), Berkely, CA, USA, 79–94.

[42] W. Lee and J. Stolfo, A framework for constructing features and models for intrusion detection systems, *ACM Transactions on Information and System Security* (*TISSEC*) **3**(4) (2000), 227–261.

[43] Y. Freund and R.E. Schapire, A decision theoretic generalization of on-line learning and an application to boosting, *Journal of Computer and System Sciences* **55**(1) (1997), 119–139.