VOL. 9, NO. 10, OCTOBER 2014 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences


www.arpnjournals.com

ADAPTIVE ANOMALY INTRUSION DETECTION SYSTEM USING OPTIMIZED HOEFFDING TREE

S. Ranjitha Kumari and Krishna Kumari P.
Department of Computer Applications, Rathnavel Subramaniam College of Arts and Science, Sulur, Coimbatore, India
E-Mail: ranjithasenthil@gmail.com

ABSTRACT

Anomaly Intrusion Detection System is used to identify a new attack in the network by identifying the deviations in the network traffic patterns. Though it identifies new attacks efficiently, the false alarm rate is usually high in this system. As there may be attack in the network at any time and as the input traffic varies over time, we need a model which efficiently identifies the change in the network traffic and adapts quickly to generate an alarm. In this paper we have proposed an adaptive anomaly intrusion detection model using stream mining approach which identifies the changes in the network and adapts the underlying model immediately. We have used optimized Hoeffding Tree where the prediction phase is optimized using Particle Swarm Optimization algorithm to increase the accuracy rate and to reduce the false alarm rate. Also the node splitting in Optimized Hoeffding Tree is controlled using error rate to keep the misclassification error rate and false alarm rate within considerable range. The results of our model are compared with the results of static intrusion detection models using unsupervised machine learning techniques. The experimental result shows that our model performed better in accuracy and false positive rate compared to the static models. We have used NSL KDD data set for our experiment.

Keywords: anomaly intrusion detection, hoeffding tree, particle swarm optimization (PSO), machine learning algorithms, NSL-KDD dataset.

INTRODUCTION

The tremendous increase in information sharing and number of users in the internet has caused the network systems more vulnerable towards cyber attacks. Hence it has become the need of the hour to protect and secure the information from such attacks. Intrusion detection system is one such technique used to secure our system from the security breaches in network. The main aim of Intrusion detection system is to detect a malicious activity in the network which compromises the integrity, confidentiality and availability of the system in the network. They can be categorized as Signature based Intrusion detection system and Anomaly based Intrusion detection system such that the former compares the network events with the already existing attack patterns and generates alarm if there is a match. The Signature based Intrusion detection system is very effective in identifying known attacks and the rate of false alarm is less. The major constraint in signature Intrusion detection system is the ability to detect new attacks as these attacks have new patterns which is not stored in the existing patterns.

Anomaly based Intrusion detection system learn the normal behavior of the network system and generate alarm if the system deviates from the normal behavior. Anomalies based Intrusion detection system is very efficient in detecting new attacks but tend generate more false alarm. The unknown activity events of legitimate user deviating from the normal network behavior will result in generation a false alarm of intrusion. There are various methods used in Anomaly Intrusion detection system such as data mining techniques, Statistical methods, Rule based techniques, machine learning techniques etc. Many static models have been proposed by the researchers to improve the accuracy of detection rate and to reduce the false alarm rate in anomaly Intrusion detection system. These static models are trained in offline mode and later they are implemented in online mode to detect the attacks. As there are new attacks in the network frequently, these static models must be updated regularly. As the traffic in the network is a continuous process, the static model cannot update the new traffic patterns and new attacks in the network. The existing model must be retrained with new patterns and again implemented.

Stream mining is an emerging field in machine learning which handles the continuous supply of data with time and memory constraints [1, 2]. The main features of stream mining are that it scans the events in the network only once and updates its model incrementally. Stream mining provides the optimal solution for the real time intrusion detection system as they can monitor the continuous network events, dynamically adapts to the changes and updates the underlying model incrementally [3]. The Hoeffding tree Algorithm is well known stream mining algorithm which is used in adaptive training model [4]. In this paper, we have proposed an Adaptive Anomaly Intrusion Detection System Model which adapts to the changes in the network traffic quickly and identifies the new attacks efficiently. Our model is built using Optimized Hoeffding Tree where the prediction phase is optimized using Particle Swarm Optimization algorithm and the node splitting in is controlled using error rate. Our model monitors the network event continuously, updates the model regularly and generates alarm when an anomalous behavior is identified in the network.

The remaining of the paper is organized in the following order: Section II describes the related work by various authors in Intrusion Detection System. Section III describes the Adaptive Anomaly Intrusion Detection

Model using optimized Hoeffding Tree where node splitting is controlled by cost of error rate and the prediction phase is optimized using PSO algorithm to improve accuracy rate. Section IV shows the experimental results and comparative analyses with other unsupervised machine learning techniques. We have given conclusion and future enhancement in Section V.

Related work

There are different methods used in Anomaly intrusion detection techniques like statistical methods, Machine learning methods, rule based methods etc. The Statistical based techniques stores the behavior of the users and uses this information for the deviation from regular behavior. In this method normal behavior data occurs in high probability regions of a stochastic model, where as anomalies occur in the low probability regions of the stochastic model [5]. For example, Frequency analysis model detects the intrusion based on frequency histogram such that an anomaly score is calculated for each packet. The fewer times a given packet seen, the higher is its anomaly score. If the Anomaly score crosses the threshold level, the alarm is raised. Multivariate model is based on correlations between two or more variables such that multivariate Intruder behavior is characterized with greater confidence by considering such correlations. A Markov process model is used to establish transition probabilities among different states. A time series model focuses on time intervals, looking for sequences of events that happen too rapidly or too slowly [1].

Machine learning techniques are widely used for Intrusion Detection Systems and they consist of computer algorithms which learn through their experience. Classification and clustering are two machine learning techniques used to identify abnormal patterns in the network. Classification techniques which are also known as supervised learning are used in signature based intrusion detection techniques to detect well known attacks. Clustering is a technique for finding patterns in an unlabelled data with many dimensions. Clustering techniques are better compared to classification techniques for anomaly intrusion detection [2]. These techniques have the ability to learn the data and detect the anomaly without prior the knowledge of intrusion patterns.

The machine learning techniques trains the model using supervised and unsupervised algorithms. The data set which is used for training requires the data to be labeled as “normal” and ‘anomaly’. A predictive model is built for normal and anomaly classes. The new incoming data are compared against this predictive model and is categorized as normal or anomaly based on this prediction. The main drawbacks in this approach are that the intrusion which appears to be normal may go undetected and to get an accurate label of an anomaly class is not easy. The algorithms which are used for building the predictive model needs to train for unlabelled patterns. Usually the false alarm rate is low in anomaly intrusion detection system when we use unsupervised machine learning techniques [6, 7] compared to supervised techniques.

Dewan Md. Farid [8] *et al*., proposed an Anomaly Network Intrusion Detection using improved Self Adaptive Bayesian Algorithm. The proposed model can process large volume of data and classifies the same with high detection speed and accuracy. They have used Bayesian classifier as the base learner such that it adjusts the weight of training examples till all the test examples are correctly classified. Farzaneh Geramiraz *et al*., [4] used Fuzzy Rule based modeling for creating the Adaptive Anomaly Intrusion Detection Model. Their model consists of four components - a Detection Model Generator, an IDS Engine, a Fuzzy Model Tuner and a Buffer. This model performs 15% higher than the static intrusion detection models.

Rangadurai Karthick R *et al*., [9] proposed an Adaptive Intrusion Detection system using two stage architecture. They have used a probabilistic classifier to detect anomalies in first stage and in second stage they have used HMM model to narrow down the potential attack IP addresses. Emma Ireland *et al*., [10] proposed two ways of training an intrusion detection system to recognize possible attacks on a system: genetic algorithms and fuzzy logic. Mohammad Sazzadul Hoque *et al*., [11] presented an Intrusion Detection System (IDS), by applying genetic algorithm (GA) to efficiently detect various types of network intrusions. Parameters and evolution processes for GA were discussed in details and implemented. Their approach uses evolution theory to information evolution in order to filter the traffic data and thus reduce the complexity. Dewan Md. Farid [12] presented a new learning algorithm for anomaly based network intrusion detection system using decision tree algorithm that distinguished attacks from normal behaviors and identifies different types of intrusions.

Hui Zhao *et al*., [13] presented a new ensemble algorithm to improve intrusion detection precision. Firstly, it generates multiple training subsets in difference by using bootstrap technology. Particle Swarm Optimization is used to optimize parameters of support vector machine in order to get base classifiers with greater difference and higher precision. Ahmed A. Elngar *et al*., [14]. They proposed PSO-Discritize-HNB IDS combines Particle Swarm Optimization (PSO) and Information Entropy Minimization (IEM) discritize method with the Hidden Naive Bayes (HNB) classifier. Shingo Mabu *et al*., [15] proposed fuzzy class-association rule mining method based on genetic network programming (GNP) for detecting network intrusions. Their proposed method can be applied to both misuse and anomaly detection in network-intrusion-detection problems.

Muhammad Qasim Ali *et al*., [16] have proposed anomaly detection model which converts a stream of input data into anomaly scores. These anomaly scores are compared with the detection threshold and further classified as normal or anomaly. Imen Brahmi *et al*., [17] have proposed a distributed Intrusion Detection system which is accurate, adaptive and extensible. They have used multiagent methodology along with data mining techniques. The multiagents are used for collecting and

analyzing the network connection and data mining techniques are used for identifying attacks.

Proposed method

Static Intrusion Detection Systems are trained first and later are implemented in network for testing. The static models are updated regularly in the off line mode and used in the intrusion detection. As the network behavior changes always and as the network is vulnerable towards the new attacks, static models are not suitable for the same [4]. Also, as the input traffic is continuous, conventional machine learning algorithms are not sufficient to handle them. The data stream emerged as a solution to handle large data. The stream mining algorithms produce model by scanning the data once, also these model are available at any time with computational and memory constraints [3]. In this paper we have proposed an Adaptive Anomaly Intrusion Detection using Optimized Hoeffding Tree which can handle the large data and is updated continuously.

We have performed our experiment using NSL-KDD data set [18, 19]. We have trained our model using NSL-KDD train data set and tested using NSL-KDD test^{+} and test^{-21} dataset in terms of accuracy and false alarm rate. We have compared our results with static intrusion detection models which uses conventional unsupervised machine learning algorithms.

Binary classification: The intrusion detection systems uses binary classifier, which analyzes the input data labeled as 'normal' or 'anomaly'. The performance of the binary classifier is evaluated based on its prediction of the classes precisely. The prediction of the classifier is compared with actual prediction of the classes. The Table-1 shows the confusion matrix of the predictions made by the classifier. The prediction classes are indicated as True Positive, False Negative, False Positive and True Negative.

Table-1. Confusion matrix.

	Predicted class positive	**Predicted class negative**
Actual class positive	True Positive (TP)	False Negative (FN)
Actual class negative	False Positive (FP)	True Negative (TN)

True positive - Prediction of class as 'normal' and actual class is 'normal'

False positive - Prediction of class as 'anomaly' and actual class is 'normal'

True negative - Prediction of class as 'normal' and actual class is 'anomaly'

False negative - Prediction of class as 'anomaly' and actual class is 'anomaly'

The performance of a good Intrusion Detection system is measured in terms of Accuracy and False Positive Rate. The ability of the system to correctly classify the input traffic as normal or anomaly is called as Accuracy rate which should be high. False alarm rate is a condition when the system generates alarm when a normal traffic is detected as an anomaly and it should be low always. The accuracy of the number of correctly classified classes is calculated using:

Accuracy = (TP + FN)/ (TP + FN + FP +TN) such that TP + FN= FP +TN =1

The no. of misclassified instances is calculated using equation:

$$C_{mis} = 1 - Accuracy$$
$$= 1 - \frac{(TP + FN)}{(TP + FN + FP + TN)}$$

The false alarm rate is calculated using:

$$C_{FAR} = \frac{FP}{FP + TN}$$

The objective of this paper is to minimize misclassification (in turn high accuracy rate) and false alarm rate; hence the total cost to minimize the error in intrusion detection system is computed using the equation:

$$C_{error} = C_{mis} + C_{FAR}$$

The minimum and maximum value of C_{mis} and C_{FAR} is 0 and 1, respectively and the mean value is 0.5. Here, the cost of error C_{error} is computed by adding the mean of C_{mis} and C_{FAR}. Hence, the best payoff and the worst payoff for C_{error} are considered as 0 and 1 respectively. The node splitting in the Hoeffding Tree model is controlled by the cost of error rate C_{error}. Particle Swarm Optimization algorithm is used to optimize the Hoeffding Tree prediction phase by minimizing the false alarm rate and the misclassification rate.

Hoeffding Tree algorithm

Stream mining is a machine learning technique which is used for continuous supply of data. The events are examined only once and the model is updated incrementally. Any change in the events is updated immediately with limited time and memory constraints. Hoeffding tree is a decision tree algorithm of stream mining which constructs decision model incrementally by examining the data only once. Hoeffding tree builds an incremental decision tree which uses Hoeffding Bound or Chernoff's bound. The leaves in the decision tree contain the class labels and the node contains the split attributes. The decision tree grows as the input data arrives and recursively replaces the leaves with the decision nodes. The sufficient information of an example with its attribute values is stored in a leaf with the class label. Once the sufficient statistics of the attributes of incoming example is accumulated in the leaf, the Hoeffding bound is used to split attributes to convert leaves into nodes. The tree grows

as the leaf is converted into a node. The data which enters through the root of the Hoeffding tree traverse through various nodes and after evaluation at every node it reaches the leaf with a class label.

In Hoeffding Tree, splitting of nodes is an important phase where a leaf is converted into nodes once sufficient statistics of the attributes of an example is accumulated. Hoeffding Tree grows by recursively replacing leaves by nodes. The node splitting is performed using information gain difference between two best attributes (x_a,x_b) and Hoeffding Bound HB. If r is the real valued random variable with range R and n is independent observation of this variable r, then the Hoeffding Bound HB states that , with probability $1-\delta$, the true mean of r is r - ε where

$$\varepsilon = \sqrt{R^2 \ln \frac{\frac{1}{\delta}}{2N}}$$

The attribute x_a is said to have better information gain than x_b, if the difference between the information gain of these attributes is more than Hoeffding Bound ε. If the information gain values of two attributes are similar and if Hoeffding Bound ε cannot decide the best attribute, then a user-defined threshold τ is used which prevents the delay in selecting the best attribute. Once the Hoeffding Bound ε becomes less than τ, the node will split at current best attribute [20].

Optimized Hoeffding Tree

In this paper we have proposed an Adaptive Intrusion Detection system using optimized Hoeffding Tree. The contribution of this paper is:

a) Optimize the prediction phase in Hoeffding Tree using Particle Swarm Optimization in order to minimize misclassification rate and false alarm rate.
b) The splitting of node at the best attribute is controlled using cost of misclassification rate and false alarm rate along with Hoeffding Bound ε and τ .

The proposed method, Optimized Hoeffding Tree is initialized using the single leaf and the data record is passed through this leaf using Hoeffding Tree. As the each example is passed through the root to the leaf using Hoeffding Tree, the sufficient statistics in the leaf is updated. These statistics has the information to grow the tree by further splitting the attributes. $n1$ specifies the count of number of examples at each leaf and the parameter n_{min} is the grace period, used for measuring the information gain of examples collected at leaf. The calculation of information gain after each and every training example is a complex task. Hence the grace period is used to decide the number of examples needed before calculating the information gain of attributes.

Let x_a be the attribute with highest information gain and x_b be the attributes with second highest information gain, then compute the difference between these two attributes $\Delta G = ((G_l)(X_l a) - ((G_l)(X_l b)$. The leaf is converted into a node with split on x_a, if $\Delta G > \varepsilon$. It becomes difficult to split the node, if the information gain of two attributes is similar. Hence a user-defined threshold τ is used, such that, if the Hoeffding Bound ε becomes less than τ , the node splits on the current best attribute irrespective of the next best attribute. In this paper we have used misclassification rate and False alarm rate to control the node splitting along with Hoeffding Bound and τ .The cost of error rate (misclassification rate and False alarm rate) controls the node splitting such that, the range of C_{error} is within 0 and 1.In the given time, the node splitting occurs in the Hoeffding tree when $\Delta G > \varepsilon$ and C_{error} is within 0 and 1, else if $\varepsilon < \tau$.

Algorithm-1 Optimized Hoeffding Tree Algorithm.

1: Let HT be a tree with a single leaf (the root)
2: **for all** training examples **do**
3: Sort example into leaf l using HT
4: *Predict class using* $PSO - NaiveBayesPrediction$
{
Compute Accuracy $= \frac{TP + FN}{TP + FN + FP + TN}$
Compute $C_{mis} = 1 - Accuracy$
Compute $C_{FAR} = \frac{FP}{FP + TN}$
Return C_{mis}
Return C_{FAR}
}
5: Update sufficient statistics in l
6: Increment n_l the number of examples seen at l
7: **if** $n_l \bmod n_{min} = 0$ **and** examples seen at l not all of same class **then**
8: Compute $\overline{G_l}(X_i)$ for each attribute
9: Let X_a be attribute with highest $\overline{G_l}$
10: Let X_b be attribute with second-highest $\overline{G_l}$
11: Compute Hoeffding bound $\epsilon = \sqrt{\frac{R^2 \ln\left(\frac{1}{\delta}\right)}{2n_l}}$
12: Calculate $C_{error} = C_{mis} + C_{FAR}$
13: **if** $X_a \neq X_\emptyset$ and $[((G_l)(X_l a) - ((G_l)(X_l b) > \epsilon$ and $(0 < C_{error} < 1)]$ or $\varepsilon < \tau$ or
14: Replace l with an internal node that splits on X_a
15: **for all** branches of the split **do**
16: Add a new leaf with initialized sufficient statistics
17: **end for**
18: **end if**
19: **end if**
20: **end for**

Prediction phase: When an event (x, y) arrives where x is the vector of d attributes and y is the class label; it is sorted from root to the leaf using Hoeffding tree algorithm. Three prediction strategies: Majority class, Naïve Bayes and Hybrid adaptive method are used to predict the class of the event in Hoeffding Tree. But in the proposed method we have used Naives Bayes classifier optimized using Particle swarm optimization to predict the classes. We have not used majority class for prediction because when a new event occurs, it predicts based on the frequent class of examples that were observed during training process. Hence it does not predict the minority class accurately and is partial towards the majority class prediction. Naives Bayes algorithm is based on the Bayesian Model with the independence of the attributes. Bayesian model is easy to build and is suitable for large datasets. Naives Bayes algorithm predicts according to the posterior probability of the class and is represented using:

$$\left(p(c)/x\right) = \frac{p(x/c)p(c)}{p(x)}$$

Particle swarm optimization is used for optimizing the rule discovery in the Naives Bayes classifier to increase the accuracy of classification. The accuracy of the prediction of the classifier is calculated using the equation:

$$\text{Accuracy} = \frac{TP + FN}{TP + FN + FP + TN}$$

And the rule coverage percentage which specifies the proportion of examples which are covered by the rule and have the class predicted by the rule is represented using:

$$\text{Rule Cover Percentage} = \frac{TP}{TP + FN + FP + TN}$$

The prediction cost of all the rules are calculated using:

$$\text{Prediction cost} = \alpha(\text{Fitness Function}) + \beta(\text{Rule cover percentage})$$ where

$$\text{Fitness function} = \frac{TP + FN}{TP + FN + FP + TN}$$

$$\text{Rule cover percentage} = \frac{TP}{TP + FN + FP + TN}$$

For an unknown example arriving, the prediction cost of all the rules is computed which covers the example. The prediction cost is evaluated and accumulated based on different classes. The class which has the highest prediction cost is selected as the final class [21].

Particle Swarm Optimization (PSO) is heuristic optimization algorithm based on swarm intelligence which is inspired by the behavior of birds or fish movement for the food. The advantages of PSO algorithm is that it is easy to implement, requires fewer parameter setting, has got faster searching speed and has low computational complexity.PSO algorithm is initialized using a random population of particles, where each particle represents a solution in the d dimensional search space [22]. In order to search the best solution, these particles move in the search space according to their own previous best position and the global best position of the swarm population. The velocity with which a particle moves in the search space is given using the equations:

$$V_{id}(t+1) = W.V_{id}(t) + c_1 r_1 (p_{id} - X_{id}(t)) + c_2 r_2 (p_{gd} - X_{id}(t))$$

$$X_{id}(t+1) = X_{id}(t) + V_{id}(t+1)$$

Where $V_{id}(t+1)$ and V_{id} (t) are the updated and current particles velocities

X_{id} (t + 1) and X_{id} (t) the updated and current particles positions.

C_1 and C_2 are two positive constants and r_1 and r_2 random numbers within the range [0, l]).

PSO algorithm

a) Initialize the swarm using i particles (each particle is the rule to predict the class of the event) with
x_{id} - current velocity of the particle
p_{id} - personal best position of the particle in search space where particle i presents the smallest error as determined by the objective function minimization task.
p_{gd} - global best position marked by represents the position yielding the lowest error amongst all the particles.
C_1 and C_2 are two positive constants and r_1 and r_2 random numbers within the range [0, l].

b) Calculate the fitness value for the each particle using fitness function
Fitness value = $\alpha(\text{Fitness Function}) + \beta(\text{Rule cover percentage})$
$\text{Fitness function} = \frac{TP + FN}{TP + FN + FP + TN}$
$\text{Rule cover percentage} = \frac{TP}{TP + FN + FP + TN}$

c) Select particles which have the best fitness value p_{id} and assign that particle position as the initial global best position p_{gd}. Best particles are selected are selected based on the minimum errors in the fitness value.

d) Update the position and velocity of the particles using equation 1 and 2

e) The fitness value of each particle is compared with its previous p_{id} value; if it is better, update the value with current p_{id} else reserve the old one.

f) The position of each particle is compared with the global best position p_{gd}, if the current position is

best, update the p_{gd} with current value else reserve the old one.

g) Terminate the process once the optimal solution is achieved.

α = [0, 1] and $\beta = [1 - \alpha]$ where alpha and beta is associated with parameters related to classification accuracy and rule coverage for the events. If α value is greater than β then the model will be sensitive towards accuracy, else the model will sensitive towards rule coverage. Hence the values should be selected carefully.

NSL-KDD DATASET

We have used NSL-KDD data set for our experiment. NSL-KDD data set is used as it solves the problem in KDD'99 training and test sets which contains huge number of redundant data. The redundant data may lead classification algorithms to be biased towards these redundant records and thus preventing it from classifying other records [18]. NSL-KDD data set are created randomly by sampling records from the #successful Prediction such that each group has an inverse proportion to the percentage of records in the original group. These train and test data sets called KDD-Train$^+$ and KDD-Test$^+$, because they contain a number of records from all groups and create new data sets. New train and test data sets include 20% of KDD-Train$^+$ and KDD-Test$^+$ data sets without any record with #successful Prediction equal to 21 [18].The generated data sets, KDDTrain$^+$ and KDDTest$^+$, includes 125, 973 and 22, 544 records, respectively. Furthermore, one more test set was generated that did not include any of the records that had been correctly classified by all 21 learners, KDDTest^{-21}, which incorporated 11, 850, records [8]. In this paper we have used KDD train data as the training data set and have tested our proposed model with KDD test$^+$ and KDD test^{-21} dataset. The NSL-KDD intrusion data set contains 41 attribute categorized into DoS (Denial of Service), R2L (Remote to Local Attack), U2R (User to Root Attack) and Probing Attack [19].

EXPERIMENTAL RESULTS

We have evaluated our experiment using MOA [7] and WEKA [23]. The unsupervised machine learning algorithms were tested using KDDTrain$^+$, KDDTest$^+$ and KDDTest^{-21} respectively. For proposed method we have used MOA [7] tool and the algorithms were trained and tested using prequential valuation method which tests and then trains the dataset. We have compared the performance of proposed model in terms of Accuracy and False alarm rate.

The accuracy percentage of correctly classified instances and false positive rate are trained and tested using NSL KDD training and testing dataset, respectively. Tables 2 and 3 shows the accuracy and false positive rate using training and test data sets, respectively.

Figure-1 and Figure-2 depicts the accuracy of correctly classified instances and false positive rate using NSL KDD Test$^+$ and KDDTest^{-21} datasets.

Table-2. Comaparision of accuracy using NSL KDD data.

Algorithm	Accuracy (%)		
	Train$^+$	KDD Test$^+$	KDD Test^{-21}
K-Means	62	68	65
Self organizing map	60	63	66
Farthest first	90	78	76
Proposed	98.20	97.1	96.2

Table-3. Comaparision of false positive rate using NSL KDD dataset.

Algorithm	False Positive rate (%)		
	Train$^+$	KDD Test$^+$	KDD Test^{-21}
K-Means	38	32	35
Self organizing map	40	37	34
Farthest first	10	22	24
Proposed	1.8	2.9	3.8

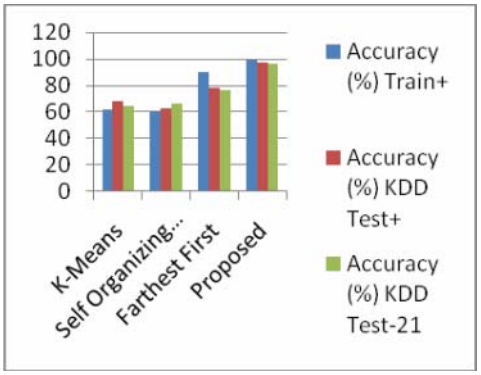

Figure-1. Accuracy Rate (%)

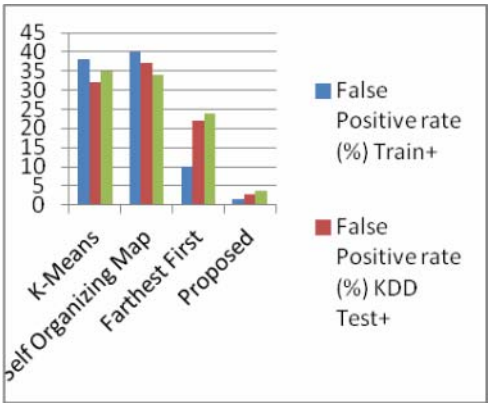

Figure-2. False Positive Rate (%)

The experimental results show that proposed model performs better in accuracy and false positive rate. The farthest first algorithm performed better compared to K-Means and SOM. The accuracy percentage was 90%, 78% and 76% for KDDTrain^{+}, KDD Test^{+} and KDDTest^{-21} respectively. Our Model had the accuracy of 98.20%, 97.10% and 96.20% for KDDTrain^{+}, KDD Test^{+} and KDDTest^{-21} respectively. The false positive rate is 1.8%, 2.9% and 3.8% KDDTrain^{+}, KDD Test^{+} and KDDTest^{-21} respectively in our model. The false positive rate is reduced to a greater extent when compared with the other unsupervised machine learning algorithms. Also, the main advantage of our model is that, it detects the changes in the network efficiently and adapts the underlying model instantly.

CONCLUSION AND FUTURE WORK

In this paper we have proposed an Adaptive Anomaly Intrusion Detection Model using stream mining concept which learns quickly and adapts easily to the changes in the network traffic. The node splitting in Hoeffding Tree is optimized using cost of error rate and the prediction phase is optimized using PSO algorithm to improve accuracy rate. We have compared the results of proposed model with the unsupervised machine learning algorithms - K-Means, SOM and Farthest First algorithms. The proposed model has got greater accuracy in classifying instances and has low false positive rate compared to unsupervised machine learning algorithms. Our future work will be to use the real time intrusion data set with different traffic patterns with many new attacks.

REFERENCES

[1] Joao Gama. 2010. Knowledge Discovery from Data Streams. Chapman and Hall/CRC.

[2] Pedro Domingos. 2000. Mining high-speed data streams. pp. 71-80. ACM Press.

[3] Albert Bifet, Geoff Holmes, Bernhard Pfahringer, Richard Kirkby and Ricard Gavald. 2009. A New ensemble methods for evolving data streams. In: Proc. of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '09, New York, NY, USA ACM. pp. 139-148.

[4] Farzaneh Geramiraz, Amir Saman Memaripour and Maghsoud Abbaspour. 2012. Adaptive Anomaly-Based Intrusion Detection System Using Fuzzy Controller. In: International Journal of Network Security. 14(6): 352-361.

[5] Kamini Nalavade and B.B. Meshram: 2013. Adaptive Anomaly Detection for Network Security. In: International Journal of Computer and Internet Security. 5(1): 1-9.

[6] Varun Chandola, Arindam Banerjee and Vipin Kumar. 2009. Anomaly detection: A survey. ACM Computing Survey. 41(3).

[7] A. Bifet, G. Holmes, R. Kirkby and B. Pfahringer. 2010. Moa: Massive online analysis. Journal of Machine Learning Research (JMLR).

[8] Rangadurai Karthick R *et al*. 2012. Adaptive network intrusion detection system using a hybrid approach. In: Proceedings: Fourth International Conference on Communication Systems and Networks (COMSNETS).

[9] Dewan Md. Farid *et al*. 2010. Anomaly Network Intrusion Detection Based on Improved Self Adaptive Bayesian Algorithm. In: Journal of computers. 5(1).

[10] Emma Ireland *et al*. 2013. Intrusion Detection with Genetic Algorithms and Fuzzy Logic. UMM CSci Senior Seminar Conference, Morris, MN.

[11] Mohammad Sazzadul Hoque *et al*. 2012. An Implementation of Intrusion Detection System Using Genetic Algorithm. International Journal of Network Security and Its Applications (IJNSA). 4(2).

[12] Dewan Md. Farid *et al*. 2010. Attacks Classification in Adaptive Intrusion Detection using Decision Tree. World Academy of Science, Engineering and Technology. p. 39.

[13] Hui Zhao *et al*. 2013. Intrusion Detection Ensemble Algorithm based on Bagging and Neighborhood Rough Set, International Journal of Security and Its Applications. 7(5): 193-204.

[14] Ahmed A. Elngar *et al*. 2013. A Real-Time Anomaly Network Intrusion Detection System with High Accuracy. Information Science Letters.

[15] Shingo Mabu *et al*. 2011. An Intrusion-Detection Model Based on Fuzzy Class-Association-Rule Mining Using Genetic Network Programming IEEE transactions on systems, man, and cybernetics-part c: applications and reviews. 41(1).

[16] Muhammad Qasim Ali *et al*. 2013. Automated Anomaly Detector Adaptation using Adaptive Threshold Tuning. In: ACM Transactions on Information and System Security (TISSEC). 15(4).

[17] Imen Brahmi *et al*. 2012. Towards a Multiagent-Based Distributed Intrusion Detection System Using Data Mining Approaches. In: Agents and Data Mining Interaction. Lecture Notes in Computer Science. 7103: 173-194.

[18] M. Tavalaee, E. Bagheri, W. Lu, and A. Ghorbani. 2009. A detailed analysis of the KDD CUP 99 data set. In: IEEE Symposium: Computational Intelligence for Security and Defense Applications, CISDA'09. pp. 1-6.

[19] nsl.cs.unb.ca/NSL-KDD/.

[20] Albert Bifet *et al*. 2011. Data stream Mining - A Practical Approach.

[21] Yu Liu *et al*. 2004. Rule Discovery with Particle Swarm Optimization AWCC 2004, LNCS 3309. pp. 291-296.

[22] Chen Wei-neng, Zhang Jun. 2010. A novel set-based particle swarm optimization method for discrete optimization problem. IEEE Transactions on Evolutionary Computation. 14(2): 278-30.0.

[23] www.cs.waikato.ac.nz/ml/weka/.