An Efficient Approach for Dynamic Distributed Network Intrusion Detection using Online Adaboost-Based Parameterized Methods

Anilkumar.V.Brahmane
G.H.Raisoni COEM
Ahmednagar
Savitribai Phule Pune University

Amruta Amune
Assistant Professor
Dept.Computer Engineering,
G.H.Raisoni COEM
Ahmednagar

ABSTRACT
Modern network intrusion detection systems are short of flexibility to the frequently altering network surroundings. Additionally, intrusion detection in the new distributed architectures is now a major requirement. In this paper, we propose online Adaboost-based intrusion detection algorithms. In an enhanced algorithm online Adaboost process and online Gaussian mixture models (GMMs) are used as weak classifiers. We further propose a distributed intrusion detection framework, in which a local parameterized detection model is created in each node using the online Adaboost algorithm. A global detection model is constructed in each node by merging the local parametric models using a small number of samples in the node. This combination is accomplished using an algorithm based on particle swarm optimization (PSO) and support vector machines. The global model in each node is used to detect intrusions. Investigational results show that the enhanced online Adaboost process with GMMs gets a superior detection rate and a lower false alarm rate than the traditional online Adaboost process that uses decision stumps. Both the algorithms outperform existing intrusion detection algorithms. It is also shown that our PSO, and SVM-based algorithm efficiently merge the local detection models into the global model in each node; the global model in a node can handle the intrusion categories that are found in other nodes, without distribution the samples of these intrusion types.

General Terms
GMM Gaussian Mixture Model, PSO SVM.

Keywords
Local Model, Global Model

1. INTRODUCTION
NETWORK ATTACK detection is one of the most important problems in network information security. Currently there are mainly firewall, network-based intrusion detection and prevention systems

(NIDS/NIPS) and unified threat management (UTM) like devices to detect attacks in network infrastructure. NIDS/NIPS detect and prevent network behaviors that violate or endanger network security. Basically, firewalls are used to block certain types of traffic to improve the security. NIDS and firewalls can be linked to block network attacks. UTM devices combine firewall, NIDS/NIPS, and other capabilities onto a single device to detect similar events as standalone firewalls and NIDS/NIPS devices. Deep packet inspection (DPI) adds analysis on the application layer, and then recognizes various applications and their contents. DPI can incorporate NIDS into firewalls. It can increase the accuracy of intrusion detection, but it is more time-consuming, in contrast to traditional package header analysis.

This paper focuses on investigation of NIDS. The current practical solutions for NIDS used in industry are misuse based methods that utilize signatures of attacks to detect intrusions by modeling each type of attack. As typical misuse detection methods, pattern matching methods search packages for the attack features by utilizing protocol rules and string matching. Pattern matching methods can effectively detect the well-known intrusions. But they rely on the timely generation of attack signatures, and fail to detect novel and unknown attacks. In the case of rapid proliferation of novel and unknown attacks, any defense based on signatures of known attacks becomes impossible. Moreover, the increasing diversity of attacks obstructs modeling signatures. This paper focuses on machine learning-based NIDS.

The machine learning-based intrusion detection methods can be classified as statistics based, data mining based, and classification based. All the three classes of methods first extract low-level features and then learn rules or models that are used to detect intrusions. A brief review of each class of methods is given below.

1) Statistics based methods construct statistical models of network connections to determine whether a new connection is an attack. For instance, Denning [1] construct statistical profiles for normal behaviors. The profiles are used to detect anomalous behaviors that are treated as attacks..

2) Data mining-based methods mine rules that are used to determine whether a new connection is an attack. For instance, Lee *et al*. [4] characterize normal network behaviors using association rules and frequent episode rules. Deviations from these rules indicate intrusions on the network. Zhang *et al*. use the random forest algorithm to automatically build patterns of attacks. Otey *et al*. [5] propose an algorithm for mining frequent itemsets (groups of attribute value pairs) to combine categorical and continuous attributes of data. .

3) Classification-based methods construct a classifier that is used to classify new connections as either attacks or normal connections. For instance, Mukkamala *et al*. use the support vector machine (SVM) to distinguish between normal network behaviors and attacks, and further identify important features for intrusion detection. Mill and Inoue propose the TreeSVM and ArraySVM algorithms for reducing the inefficiencies that arise when a sequential minimal optimization algorithm for intrusion detection is learnt from a large set of training data. Zhang and Shen [8] use SVMs to implement online intrusion detection. Kayacik *et al*. [6] propose an algorithm for intrusion detection based on the Kohonen self-organizing feature map (SOM).

2. RELATED WORK

J. B. D. Caberera, B. Ravichandran, and R. K. Mehra [2]- Examines the application of statistical traffic modeling for detecting novel attacks against computer networks. In this paper it is discuss the application of network activity models and application models using the 1998 DARPA Intrusion Detection Evaluation data set. Network activity models monitor the volume of traffic in the network, while application models describe the operation of application protocols.

W. Lee, S. J. Stolfo, and K. Mork[3]- This paper describe a data mining framework for adaptively building Intrusion Detection (ID) models. The central idea is to utilize auditing programs to extract an extensive set of features that describe each network connection or host session, and apply data mining programs to learn rules that accurately capture the behavior of intrusions and normal activities. These rules can then be used for misuse detection and anomaly detection.

H. G. Kayacik, A. N. Zincir-heywood, and M. T. Heywood[6]- An approach to network intrusion detection is investigated, based purely on a hierarchy of Self-Organizing Feature Maps. Our principle interest is to establish just how far such an approach can be taken in practice. To do so, the KDD benchmark dataset from the International Knowledge Discovery and Data Mining Tools Competition is employed.

J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas[12]- This paper describe a common theoretical framework for combining classifiers which use distinct pattern representations and show that many existing schemes can be considered as special cases of compound classification where all the pattern representations are used jointly to make a decision.

P. Z. Hu and M. I. Heywood[7]- Intrusion Detection Systems are typically deployed for real time operation, but are limited to identifying attacks once initiated. In this work we instead investigate the potential for predicting an attack before it occurs. To do so, a two-stage process is employed with a classification stage following that of a predictor. Predictors are based on the SOM and classifier on an SVM.

Z. Zhang and H. Shen[8]- To break the strong assumption that most of the training data for intrusion detectors are readily available with high quality, conventional SVM, robust SVM and one-class SVM are modified respectively in virtue of the idea from online support vector machine (OSVM) in this paper, and their performances are compared with that of the original algorithms.

H. Lee, Y. Chung, and D. Park [9]- An adaptive intrusion detection algorithm which combines the Adaptive Resonance Theory(ART) with the Concept Vector and the Mecer-Kernel is presented. Compared to the supervised- and the clustering-based Intrusion Detection Systems (IDSs), our algorithm can detect unknown types of intrusions in on-line by generating clusters incrementally.

3. CHALLENGES IN EXITING SYSTEM

1)Network environments and the intrusion training data change rapidly over time, as new types of attack emerge. In addition, the size of the training data increases over time and can become very large. Most existing algorithms for training intrusion detectors are offline. The intrusion detector must be retrained periodically in batch mode in order to keep up with the changes in the network. This retraining is time consuming.

2) There are various types of attributes for network connection data, including both categorical and continuous ones, and the value ranges for different attributes d iffer greatly—from *{*0, 1*}* to describe the normal or error status of a connection, to [0, 107] to specify the number of data bytes sent from source to destination. The combination of data with different attributes without loss of information is crucial to maintain the accuracy of intrusion detectors.

3) In traditional centralized intrusion detection, in which all the network data are sent to a central site for processing, the raw data communications occupy considerable network bandwidth. There is a computational burden in the central site and the privacy of the data obtained from the local nodes cannot be protected.

4. THE PROPOSED SYSTEM

This project addresses the above challenges and proposes a classification based framework for the dynamic distributed network intrusion detection using the online Adaboost algorithm. The Adaboost algorithm is one of the most popular machine learning algorithms. Its theoretical basis is sound, and its implementation is simple.

Moreover, the Adaboost algorithm corrects the misclassifications made by weak classifiers and it is less susceptible to over-fitting than most learning algorithms. Recognition performances of the Adaboost-based classifiers are generally encouraging. In this project , a hybrid of online weak classifiers and an online Adaboost process results in a parameterized local model at each node for intrusion detection.

The parametric models for all the nodes are combined into a global intrusion detector in each node using a small number of samples, and the combination is achieved using an algorithm based on particle swarm optimization (PSO) and SVMs. The global model in a node can handle the attack types that are found in other nodes, without sharing the samples of these attack types.

Solution suggested in proposed system.

1) In the Adaboost classifier, the weak classifiers are constructed for each individual feature component, for both continuous and categorical ones, in such a way that the relations between these features can be naturally handled, without any forced conversions between continuous features and categorical features.

2) New algorithms are designed for local intrusion detection. The traditional online Adaboost process and a newly proposed online Adaboost process are applied to construct local intrusion detectors. The weak classifiers used by the traditional Adaboost process are decision stumps. The new Adaboost process uses online Gaussian mixture models (GMM) as weak classifiers. In both cases the local intrusion detectors can be updated online. The parameters in the weak classifiers and the strong classifier construct a parametric local model.

3) The local parametric models for intrusion detection are shared between the nodes of the network. The volume of communications is very small and it is not necessary to share the private raw data from which the local models are learnt.

4) We propose a PSO and SVM-based algorithm for combining the local models into a global detector in each node. The global detector that obtains information from other

nodes obtains more accurate detection results than the local detector.

5. OVERVIEW OF PROPOSED SYSTEM

Data Preprocessing: For each network connection, three groups of features that are commonly used for intrusion detection are extracted: basic features of individual transmission control protocol (TCP) connections, content features within a connection suggested by domain knowledge, and traffic features computed using a two-second time window.

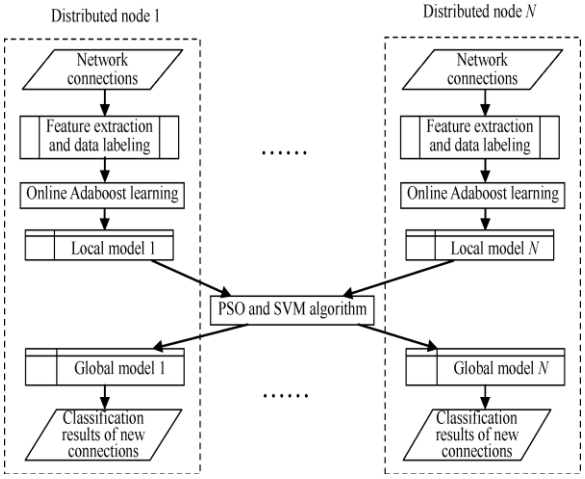

Fig.1. Overview of the intrusion detection framework

The extracted feature values from a network connection form a vector $x = (x1, x2, \ldots, xD)$, where D is the number of feature components. There are continuous and categorical features, and the value ranges of the features may differ greatly from each other. The framework for constructing these features can be found in [9]. A set of data is labeled for training purposes. There are many types of attacks on the Internet. The attack samples are labeled as −1, −2, ... depending on the attack type, and the normal samples are all labeled as +1.

Local Models: The construction of a local detection model at each node includes the design of weak classifiers and Adaboost-based training. Each individual feature component corresponds to a weak classifier. In this way, the mixed attribute data for the network connections can be handled naturally, and full use can be made of the information in each feature. The Adaboost training is implemented using only the local training samples at each node. After training, each node contains a parametric model that consists of the parameters of the weak classifiers and the ensemble weights.

Global Models: By sharing all the local parametric models, a global model is constructed using the PSO and SVM-based algorithm in each node. The global model in each node fuses the information learned from all the local nodes using a small number of training samples in the node. Feature vectors of new network connections to the node are input to the global classifier, and classified as either normal or attacks. The results of the global model in the node are used to update the local model in the node and the updated model is then shared by other nodes.

6. ALGORITHM

Local Detection Model

We apply GMM & online Adaboost to construct the local intrusion detection models.

The Online GMM (Gaussian Mixture Models)

For the samples of each attack type or normal samples, we use a GMM to model the data on each feature component. Let $c \in \{+1,-1,-2, \ldots,-M\}$ be a sample label, where +1 represents the normal samples and "−1,−2,. . .−*M*" represents different types of attacks where M is the number of attack types.

The GMM model $\theta^c j$ on the jth feature component for the samples with label c is represented as,

$$\theta_j^c = \left\{\omega_j^c(i), \mu_j^c(i), \sigma_j^c(i)\right\}_{i=1}^{K}$$

where K is the number of GMM components indexed by i, and ω, μ, and σ represent the weight, mean, and standard deviation for the corresponding component. Then, the weak classifier on the jth feature is constructed as,

$$h_j(x) = \begin{cases} 1 & if \quad p(x|\theta_j^{+1}) > \frac{1}{M} p(x|\theta_j^c) \\ & for \quad c = -1, -2, \ldots, -M \\ -1 & \text{otherwise} \end{cases}$$

where "1/M" is used to weight the probabilities for attack types in order to balance the importance of the attack samples and the normal samples.

In this way the sum of the weights for all the types of attack samples is equal to the weight of normal samples, and the false alarm rate is reduced for the final ensemble classifier.

New Online Adaboost Algorithm.

Our new online Adaboost algorithm selects a number of weak classifiers according to a certain rule and updates them simultaneously for each input training sample. Let $S+$ and $S-$ be, respectively, the numbers of the normal and attack samples that have already been input. Let be the number of the samples, each of which is correctly classified by the previous strong classifier that has been trained before the sample is input.

Let S^+ and S^- be, respectively, the numbers of the normal and attack samples.

Let Ω be the number of the samples.

Let λ_t^{sw} be the sum of the weights of the input samples.

Let Ct be the number of the input samples.

For a new sample (x,y),y belong to {1,-1,-2,…},the strong classifier is updated by the following steps,

Step 1: Update the parameter $S+$ or $S-$ by

$$\begin{cases} S^+ \leftarrow S^+ + 1 & \text{if } y = 1 \\ S^- \leftarrow S^- + 1 & \text{else} \end{cases}$$

Initialize the weight λ of the new sample by,

$$\begin{cases} \lambda = (S^+ + S^-)/S^+ & \text{if} \quad y = 1 \\ \lambda = (S^+ + S^-)/S^- & \text{else} \end{cases}$$

where λ follows the change of $S+$ and $S-$, in favor of balancing the proportion of the normal samples and the attack samples to ensure that the sum of the weights of the attack samples equals to the sum of the weights of the normal samples.

Step 2: Calculate the combined classification rate υt for each weak classifier *ht* by

$$\upsilon_t = (1-\rho)\varepsilon_t - \rho \text{sign}(y)h_t(x)$$
$$= (1-\rho)\frac{\lambda_t^{sw}}{\lambda_t^{sc}+\lambda_t^{sw}} - \rho \text{sign}(y)h_t(x)$$

Where p is a weight ranging in (o,0.5]. The rate υt combines the historical false classification rate εt of *ht* for samples input previously and the result of *ht* for the current sample (x,y).

The rate υt is more effective than εt, as it gives *ht* whose εt is high more chance to be updated and then increases the detection rate of *ht* .

The weak classifiers {*ht*} *t*=1,...,*D* are ranked in the ascending order of υt and then the weak classifiers are represented by {*hr*1, *hr*2, . . . , *hrD* }, *ri* ∈ {1, 2, . . . , *D*}.

Step 3: The weak classifies whose combined classification rates υri are not larger than 0.5 are selected from {*hr*1, *hr*2, . . . , *hrD* }. Each of these selected weak classifiers *hri* is updated using (*x*, *y*) in the following way. Compute the number *Pi* of iterations for *hri* by

$$Pi = Integer(P \exp(-\gamma(\upsilon i - \min \upsilon)))$$

where γ is an attenuation coefficient and *P* is a predefined maximum number of iterations.

Repeat the following steps (a) and (b) *Pi* times in order to update *hri*:

(a) Online update *hri* using (*x*, *y*).

(b) Update the sample weight λ, $\lambda^{sc} t$, and $\lambda^{sw} t$:

Step 4: Update *Ct* , $\lambda^{sc} t$ and $\lambda^{sw} t$ for each *ht* of the weak classifiers whose combined classification rates are larger than 0.5 ($\upsilon t > 0.5$):

Step 5: Update the parameter Ω : If the current sample is correctly classified by the previous strong classifier that has been trained before the current sample (x, y) is input, then Ω ← Ω+ 1.

Step 6: Construct the strong ensemble classifier. Calculate the ensemble weight $\alpha^* t$ of h_t by

$$\alpha_t^* = \beta \log\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right) + (1-\beta)\log\left(\frac{C_t}{\Omega}\right)$$

The $\alpha^* t$ is normalized to α_t . by

$$\alpha_t = \frac{\alpha_t^*}{\sum_{i=1}^{D}\alpha_i^*}.$$

The strong classifier H(x) is defined by

$$H(x) = \text{sign}(\sum_{t=1}^{D}\alpha_t h_t(x)).$$

The term log(*Ct*/Ω), which is called a contributory factor, represents the contribution rate of *ht* to the strong classifier. It can be used to tune the ensemble weights to attain better detection performance.

Local Parameterized Model

Subsequent to the construction of the weak classifiers and the online Adaboost learning, a local parameterized detection model ϕ is formed in each node. The local model consists of the parameters ϕw of the weak classifiers and the parameters ϕd for constructing the Adaboost strong classifier: $\phi = \{\phi w, \phi d\}$.

The parameters for each GMM-based weak classifier include a set of GMM parameters $\varphi w = \{\theta c\ j\ |j = 1, 2, ...,D; c = 1,-1,-2, ...\}$. The parameters of the strong classifier for the online Adaboost algorithm include a set of ensemble weights $\varphi d = \{\alpha t\ |t = 1, 2, ...,D\}$ for the weak classifiers.

The parameters in the GMM-based weak classifiers depend on the distributions of the different types of attacks and normal behaviors in each component of the feature vectors. The parameters in the strong classifier depend on the significances of individual feature components for intrusion detection. The local detection models capture the distribution characteristics of observed mixed attribute data in each node. They can be exchanged between the different nodes.

The parametric models are not only concise to be suitable for information sharing, but also very useful to generate global intrusion detection models.

Global Detection Models

The local parametric detection models are shared among all the nodes and combined in each node to produce a global intrusion detector using a small number of samples left in the node. This global intrusion detector is more accurate than the local detectors that may be only adequate for specific attack types, due to the limited training data available at each node.

We combine the PSO and SVM algorithms, in each node, to construct the global detection model. We use

the PSO to search for the optimal local models and the SVM is trained using the samples left in a node. Then, the trained SVM is used as the global model in the node. By combining the searching ability of the PSO and the learning ability of the SVM for small sample sets, a global detection model can be constructed effectively in each node.

The state of a particle *i* used in the PSO for one of the *A* nodes is defined as *Xi* = (*x*1, *x*2, . . . , *xA*), *xj* ∈ {0, 1}, where "*xj* = 1" means that the *j*th local model is chosen, and "*xj* = 0" means the *j*th local model is not chosen. For each particle, a SVM classifier is constructed. Let *L* be the number of local detection nodes chosen by the particle state *Xi*. For each network connection sample left in the node, a vector (*r*1, *r*2, . . . , *rL*) is constructed, where *rj* is the result of the *j*th chosen local detection model for the sample.

For each particle *i*, its individual best state *Sl i* , which has the maximum fitness value, is updated in the *n*th PSO iteration using the following equation:

$$S_i^l = \begin{cases} X_{i,n} & \text{if } f(X_{i,n}) > f(S_i^l) \\ S_i^l & \text{else} \end{cases}$$

The global best state *Sg* in all the particles is updated by

$$S^g = \arg\max_{S_i^l} f(S_i^l)$$

Each particle in the PSO is associated with a velocity that is modified according to its current best state and the current best state among all the particles.

$$V_{i,n+1} = F\left(wV_{i,n} + c_1\mu_1(S_i^l - X_{i,n}) + c_2\mu_2(S^g - X_{i,n})\right)$$

where w is the inertia weight that is negatively correlated with the number of iterations; $c1$ and $c2$ are acceleration constants called the learning rates; $\mu 1$ and $\mu 2$ are relatively independent random values in the range [0, 1]; and $F()$ is a function that confines the velocity within a reasonable range.

$$F(V) = \begin{cases} V & \text{if } \|V\| \le V_{\max} \\ V_{\max} & \text{else} \end{cases}$$

The SVM- and PSO-based fusion algorithm is outlined as follows.

Step 1: Initialization: The particles $\{X_{i,0}\}^Q_{i=0}$ are randomly chosen in the particle space, where Q is the number of particles. $S^l\ i = X_{i,0}$. A SVM classifier is constructed for each particle, and the detection rate $\gamma(X_{i,0})$ of the SVM classifier is calculated. The fitness value $f(X_{i,0})$ is calculated.

Step 2: The velocities $\{V_{i,n+1}\}^Q_{i=1}$ are updated.

Step 3: The particles' states $\{X_{i,n+1}\}^Q_{i=1}$ are evolved.

Step 4: The SVM classifier is constructed for each particle, and the detection rate $\gamma(Xi,n+1)$ is calculated.

The fitness values $f(Xi,n+1)$ are calculated.

The individual best states $\{S^l_i\}^Q_{i=1}$ are updated.

The global best sate Sg is updated.

Step 5: $n \leftarrow n + 1$. If $f(Sg) >$ *max fintness* or the predefined number of iterations is achieved, then the particle evolution process terminates and the SVM classifier corresponding to Sg is chosen as the final classifier—the global model in the node; otherwise go to Step 2 for another loop of evolution.

When certain conditions are met, nodes may transmit their local models to each other. Then, each node can construct a customized global model using a small set of training samples randomly selected from the historical training samples in the node according to the proportion of various kinds of the network behaviors. Once local nodes gain their own global models, the global models are used to detect intrusions; for a new network connection, the vector of the results from the local models chosen by the global best particle is used as the input to the global model whose result determines whether the current network connection is an attack.

7. RESULT ANALYSIS

We use the knowledge discovery and data mining (KDD) CUP 1999 dataset to test our algorithms.

Attacks in the dataset fall into the following four main categories.

- DOS: denial-of-service.
- R2L: unauthorized access from a remote machine, e.g., guessing password.
- U2R: unauthorized access to local super-user (root) privileges.
- Probe: surveillance and other probing, e.g., port scanning.

Table 1 The KDD Cup 1999 Dataset

Categories	Training data	Test data
Normal	97278	60593
DOS	391458	223298
R2L	1126	5993
U2R	52	39
Probing	4107	2377
Others	0	18729
Total	494021	311029

Predataset

Preprocessing on this input dataset will remove the 19 entries and gives us the new predataset.

The predataset is the random collection of packets data captured from each connection. The entries in predataset are difficult to read and understand. So features of each captured packet is extracted.

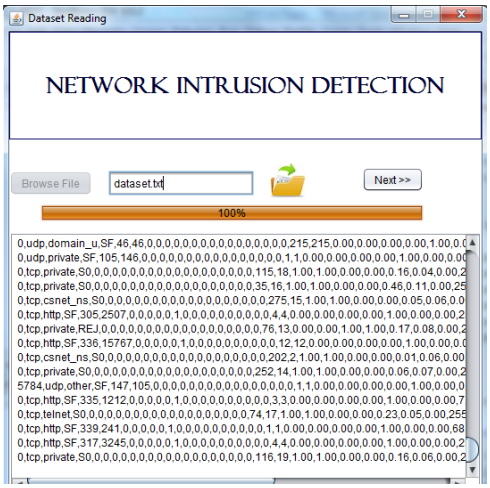

Fig.2. Predataset Feature Extracion

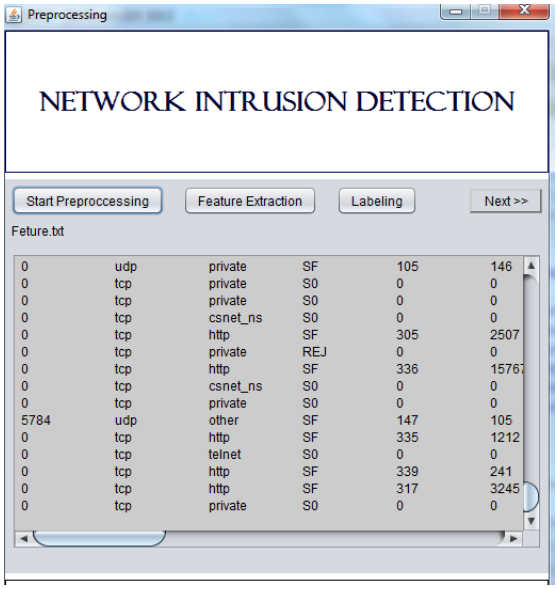

Fig.3.Features Extraction

Labeling
Labeling is done on each captured packet to recognize the normal and anomaly behavior of the connections. +1 indicates the normal and -1 indicate the attacks.

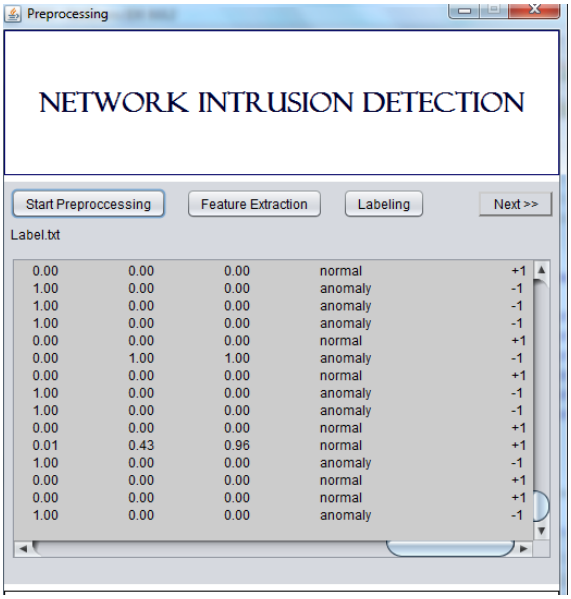

Fig.4. +1, -1 Labels indicates the normal and anomaly attack

Table 2 shows, respectively, the results of the classifier with decision stumps and the traditional online Adaboost and the classifier with online GMMs and our online Adaboost, when only continuous features or both continuous features and categorical features are used, respectively, tested on the KDD training and test datasets, respectively. It is seen that the results obtained by using both continuous and categorical features are much more accurate than the results obtained by only using continuous features. This shows the ability of our algorithms to handle mixed features in network connection data

Table 2 Results Obtained by Using Only Continuous Features or Both Continuous and Categorical Features

Algorithm	Features	Training data		Test data	
		Detection rate (%)	False Alarm rate	Detection rate (%)	False Alarm rate
Decision stumps +Traditional online Adaboost	Only continuous	98.68	8.35	90.05	13.76
	Continuous +Categorical	98.93	2.37	91.27	8.38
Online GMM + Our online Adaboost	Only continuous	98.79	7.83	91.33	11.34
	Continuous +Categorical	99.02	2.22	92.66	2.67

Successive results are still expected, when the weak classifiers are collected to obtain strong classifier. These strong classifiers at each node propagate the Local detection model. Local detection model at each node is used to form the Global detection model using PSO-SVM algorithm. This Global detection model is shared among the number of nodes which is used to detect the intrusion effectively & efficiently.

8. ACKNOWLEDGMENTS
I would like to take this opportunity to express my heartfelt thanks to my guide Prof.Amruta Amune for her esteemed guidance and encouragement, especially through difficult times. Her suggestions broaden my vision and guided me to succeed in this work. I am also very grateful for her guidance and comments while designing part of my research paper andlearnt many things under her leadership.

9. CONCLUSION
In this paper, we proposed online Adaboost-based intrusion detection algorithms, online GMMs were used as weak classifiers for our proposed online Adaboost. The results of the algorithm using decision stumps and the traditional online Adaboost were compared with the results of the algorithm using online GMMs and our online Adaboost. We further proposed a distributed intrusion detection framework, in which the parameters in the online Adaboost algorithm formed the local detection model for each node, and local models were combined into a global detection model in each node using a PSO and SVM-based algorithm. The advantages of our work are as follows: 1) our online Adaboost-based algorithms successfully overcame the difficulties in handling the mixed attributes of network connection data; 2) the online mode in our algorithms ensured the adaptability of our algorithms to the changing environments; the information in new samples was incorporated online into the classifier, while maintaining high detection accuracy; 3) our local parameterized detection models were suitable for information sharing: only a very small number of data were shared among nodes; 4) no original network data were shared in the framework so that the data privacy was protected; and 5) each global detection model improved considerably on the intrusion detection accuracy for each node.

9. REFERENCES
[1] Weiming Hu, Jun Gao, Yanguo Wang, Ou Wu, and Stephen Maybank, " Online Adaboost-Based Parameterized Methods for Dynamic Distributed Network Intrusion Detection" IEEE TRANSACTIONS ON CYBERNETICS, VOL. 44, NO. 1, JANUARY 2014

[2] D. Denning, "An intrusion detection model," *IEEE Trans. Softw. Eng.*, vol. SE-13, no. 2, pp. 222–232, Feb. 1987

[3] J. B. D. Caberera, B. Ravichandran, and R. K. Mehra, "Statistical traffic modeling for network intrusion detection," in *Proc. Modeling, Anal. Simul. Comput. Telecommun. Syst.*, 2000, pp. 466–473.

[4] W. Lee, S. J. Stolfo, and K. Mork, "A data mining framework for building intrusion detection models," in *Proc. IEEE Symp. SecurityPrivacy*, May 1999, pp. 120–132.

[5] M. E. Otey, A. Ghoting, and S. Parthasarathy, "Fast distributed outlier detection in mixed-attribute data sets," *Data Ming Knowl. Discovery*,vol. 12, no. 2–3, pp. 203–228, May 2006.

[6] H. G. Kayacik, A. N. Zincir-heywood, and M. T. Heywood, "On the capability of an SOM based intrusion detection system," in *Proc. Int.Joint Conf. Neural Netwo.*, vol. 3. Jul. 2003, pp. 1808–1813.

[7] P. Z. Hu and M. I. Heywood, "Predicting intrusions with local linear model," in *Proc. Int. Joint Conf. Neural Netw.*, vol. 3, pp. 1780–1785,Jul. 2003.

[8] Z. Zhang and H. Shen, "Online training of SVMs for real-time intrusion detection," in *Proc. Adv. Inform. Netw. Appl.*, vol. 2, 2004, pp. 568–573.

[9] H. Lee, Y. Chung, and D. Park, "An adaptive intrusion detection algorithm based on clustering and kernel-method," in *Proc. Int. Conf.Adv. Inform. Networking Appl.*, 2004, pp. 603–610.

[10] W. Lee and S. J. Stolfo, "A framework for constructing features and models for intrusion detection systems," *ACM Trans. Inform. Syst.Security*, vol. 3, no. 4, pp. 227–261, Nov. 2000.

[11] A. Fern and R. Givan, "Online ensemble learning: An empirical study," in *Proc. Int. Conf. Mach. Learning*, 2000, pp. 279–286.

[12] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 3, pp.226–238, Mar. 1998.

[13] J. Kennedy, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, 1995, pp. 1942–1948.

[14] Y. Shi and R. C. Eberhart, "A modified particle swarm optimizer," in *Proc. IEEE Int. Conf. Evolut. Comput.*, 1998, pp. 69–73.

[15] S. Stofo *et al. The Third International Knowledge Discovery and Data Mining Tools Competition,* The University of California, 2002 [Online].Available: http://kdd.ics.uci.edu/databases/kddCup99/kddCup99.h-tml.

[16] S. Mukkamala, A. H. Sung, and A. Abraham, "Intrusion detection using an ensemble of intelligent paradigms," *Netw. Comput. Appl.*, vol. 28, no. 2, pp. 167