October 2014, Volume 1, Number 4 (pp. 293–305)
http://www.jcomsec.org

Journal of Computing and Security

Effective Intrusion Detection with a Neural Network Ensemble Using Fuzzy Clustering and Stacking Combination Method

Mohammad Amini [a,*]
Jalal Rezaeenoor [a]
Esmaeil Hadavandi [b]
[a] *Department of Information Technology, University of Qom, Qom, Iran.*
[b] *Department of Industrial Engineering, AmirKabir University of Technology, Tehran, Iran.*

A R T I C L E I N F O.

Article history:
Received: 29 March 2014
Revised: 28 September 2014
Accepted: 20 November 2014
Published Online: 27 February 2015

Keywords:
Intrusion Detection, Ensemble Classifiers, Stacking, Fuzzy Clustering, Artificial Neural Networks

A B S T R A C T

Data mining techniques are widely used for intrusion detection since they have the capability of automation and improving the performance. However, using a single classification technique for intrusion detection might involve some difficulties and limitations such as high complexity, instability, and low detection precision for less frequent attacks. Ensemble classifiers can address these issues as they combine different classifiers and obtain better results for predictions. In this paper, a novel ensemble method with neural networks is proposed for intrusion detection based on fuzzy clustering and stacking combination method. We use fuzzy clustering in order to divide the dataset into more homogeneous portions. The stacking combination method is used to aggregate the predictions of the base models and reduce their errors in order to enhance detection accuracy. The experimental results on NSL-KDD dataset demonstrate that the performance of our proposed ensemble method is higher compared to other well-known classification techniques, particularly when the classes of attacks are small.

1 Introduction

The use of Internet has been expanded in many areas of daily life as well as business activities. Organizations usually employ complex information networks and open them in order to share information with their partners and customers. Therefore, information security and networks protection need to be carefully taken care of. However, as technology is progressing, the security attacks and intrusions are also developing. Nowadays, it is obvious that the basic security measures and tools such as firewalls or anti-viruses are not robust enough to provide the adequate level of security for corporate networks and prevent malicious users from breaking into the systems. A more sophisticated and intelligent solution for preventing intrusion and malicious activities is provided by Network Intrusion Detection Systems (NIDS). A NIDS is a tool which monitors and analyses data traffic on the network to detect possible attacks or intrusive activities.

Generally, intrusion detection can be interpreted as a classification problem in which a given network event is classified as normal or intrusive [1]. Intrusion detection methods are basically divided into 2 categories: misuse detection and anomaly detection [2]. In misuse detection, a model is created based on known attack

* Corresponding author.
Email addresses: m.amini84@gmail.com (M.Amini), j.rezaee@qom.ac.ir (J. Rezaeenoor), es.hadavandi@aut.ac.ir (E. Hadavandi)
ISSN: 2322-4460 © 2014 JComSec. All rights reserved.

types or intrusion scenarios. The network events are compared against this model to find matches with known attacks patterns. Usually the network events are either recognized as Normal or assigned to one of the four attack types DoS, Probe,U2R, and R2L.This method can be really effective in detecting known attacks. However, it has many limitations for detecting novel or unknown threats. In anomaly detection, a profile of the normal network traffic is generated first. The idea is to analyze the network events against this normal profile. If any deviation from the normal behavior is detected, it can be identified as intrusive. The major advantage of this method is the ability to recognize unseen attacks. However, this method can create a considerably high rate of false alarms.

Recently, machine learning and data mining techniques have attracted high attention for intrusion detection in both mentioned categories [1, 3]. The major advantage offered by these methods is their generalization ability that enables them to detect not only known threats but also their variations. The goal in intrusion detection methods is to maximize detection accuracy and reduce false positive rate as much as possible. Mostly, researchers have developed classification models for intrusion detection using single classifiers such as decision tree [4], support vector machine (SVM) [5, 6], artificial neural networks (ANNs) [7, 8], and K-nearest neighbor (KNN) [9, 10]. Although these methods have been successful in detection to some degree, they suffer from a number of limitations. Especially, in misuse detection, they have a low detection precision for low frequent attacks since they lack a sufficient number of samples to learn the rare attacks patterns. Another problem is that training the classifiers can be time consuming and create very complex models because the dataset is very large. The other issue is the lack of stability. For example the ANN is unstable and may converge to the local minimum [11].We need to have stable models which detect the known and unknown attacks accurately with a high degree of certainty.

The ensemble classifiers aim to improve the performance of classification by combining the output of multiple relatively-weak classifiers [12]. These weak classifiers are trained by different samples and develop different models. Then a combination strategy, such as majority vote, mixes their individual predictions so that they will generate better results. This way, the performance of the whole system is improved. These methods are very flexible and able to produce more accurate and stable results. In this article, we propose an ensemble classifier for intrusion detection which divides the problem space by fuzzy clustering method, and then utilizes the generated subspaces to train similar ANNs. To aggregate the predictions of all individual classifiers we apply stacked generalization in which another ANN is used to aggregate the outputs and reduce the errors of individual classifiers. We use a hybrid combination strategy in order to improve the classification accuracy. By using this ensemble system, the major drawbacks of the current single classifiers are removed. We also used radial basis function (RBF) neural networks which are more precise than multi-layer perceptron (MLP) networks in terms of generalization error. The results of our experiments are compared against other well-known classifiers such as decision tree, KNN and Naïve Bayes, as well as two types of popular ensemble methods named Boosting and Bagging. The experimental results on NSL-KDD, which is a modified version of the popular KDD99 dataset, demonstrate better performance on low frequent attacks as well as the whole classification task compared to the aforementioned methods.

The remainder of this paper is organized as follows: In the next section, the fundamentals of ensemble classification techniques and the related works for intrusion detection are described. In Section 3, we describe the framework of our proposed ensemble system and explain its working modules. Section 4 presents the experimental procedures, evaluation criteria, and discussion. Finally, Section 5 draws conclusion and describes future work.

2 Ensemble Methods for Intrusion Detection

Ensemble learning has been one of the most popular approaches in complex classification tasks in recent years. This classification method focuses on the idea that by combining several individual classifiers, it is possible to acquire a classifier which has a higher performance than that of all the individual classifiers [12]. The predictions of individual classifiers (or ensemble members) are combined using a proper method, e.g. voting or averaging, to produce a final prediction. The benefits given by the combination of redundant and complementary classifiers are increasing accuracy, robustness and overall generalization capability in most applications [13].

Building an ensemble system consists of three phases: 1) Data sampling and selection, which creates diversity among ensemble members; 2) Training member classifiers which is performed using numerous competing algorithms such as bagging, boosting, stacked generalization, etc. 3) combining member predictions using different rules e.g. majority voting, simple or weighted average [14]. Many previous research works have shown that an ensemble classifier is often more precise and robust than all its member classifiers if the individual members perform well to some degree and have independent predictions [15–19].

Considering the benefits which ensemble systems bring to the world of pattern recognition and classification, some researchers began to use these methods in intrusion detection context. Chauhan et al. compared the performance of ten best selected classifiers including C4.5, BayesNet, Logistic Regression, Stochastic Gradient Descent (SGD), Instance Based Knowledge(IBK) , JRip, PART, Random Forest, Random Tree and REPTree [20]. The authors' experiments showed that Random Forest, which is an ensemble of decision tree classifiers, outperformed other methods with respect to accuracy, specificity and sensitivity. Several ensemble methods have been proposed for misuse and anomaly detection. Table 1 summarizes the most important research works on this area and the elements of creating their ensemble classifiers.

Mukkamala et al. demonstrated the advantage of using ensemble approaches for intrusion detection. They created an ensemble of SVM, MARS and ANN classifiers and their results showed the obvious superiority of the ensemble system against the base classifiers in each of the 5 classes [21]. Giacinto et al. trained the individual classifiers with different feature subsets and obtained better performance in misuse detection after combining them [22]. For anomaly detection, Giacinto established an ensemble approach with a modular multiple classifier system using the base unsupervised classifiers. Each classifier was trained on a group of similar network protocols or services [23].

Some authors tried to use slightly different methods for creating ensembles. In [24] classifiers for misuse detection were combined serially and the network traffic was inspected in sequential stages. A hybrid intrusion detection method was proposed by Kim et al. which hierarchically integrated a misuse detection model and an anomaly detection model [25].The authors used C4.5 decision tree (DT) to create the misuse detection model and applied that model to decompose the normal training data into smaller subsets. Then, the one-class support vector machine (1-class SVM) was used to create an anomaly detection model in each decomposed region. Their experiments demonstrated that the proposed hybrid intrusion detection method could improve the IDS in terms of detection performance for unknown attacks and detection speed. Some methods make use of hybrid approaches in ensemble classifiers. Chebrolu et al. developed an ensemble system by combining Bayesian Networks (BN) and Classification and Regression Trees (CART) which performed the classification task after feature reduction [26]. Then they created a hybrid architecture with the ensemble and the base classifiers themselves to build a lightweight IDS. Their results were significant, especially for smaller classes i.e. U2R and R2L . In [27] the authors combined the hybrid systems with the ensembles. They made a hybrid system using two classification techniques DT and SVM and then developed an ensemble classifier through combining the hybrid system and the base classifiers. They were able to produce the best accuracy in detection for all 5 classes. Particularly, the detection rate for Probe attacks was 100%. Abraham and Jain applied DT, LGP and fuzzy rule sets to create an ensemble model [28]. They demonstrated that different models offer complementary information about examples' class labels. The authors selected the best model for each class as the classifier responsible for that class and then combined them with an efficient method.

One of the best methods which can provide good accuracy and flexibility in pattern recognition and data classification is using artificial neural networks (ANNs). They are good candidates as base classifiers in ensemble models. Hansen and Salamon demonstrated that the generalization ability for an ensemble of neural networks can be significantly high [29]. ANN ensembles have been used in many applications by various authors [30–34]. However, few studies have been done about the ensemble of ANNs and their performance for intrusion detection. Ghadiri and Ghadiri proposed a two-layer hybrid architecture for intrusion detection using fuzzy clustering and artificial neural networks [35]. The first layer used FCM and GK fuzzy clustering to extract the features and the second layer used a set of neural networks to perform the classification. Their result showed high detection and low false positive rates. However, the authors mostly concentrated on the clustering stage and didn't focus on the concept of ensemble learning. Thus, there was no clear method for combining the individual classifiers.

In this article, we propose an ensemble model using Radial Basis Function (RBF) and Multi-Layer Perceptron (MLP) neural networks. By using RBF networks as the base classifiers we are much less likely to deal with the problem of getting stuck in the local minima. Moreover, we aggregate the predictions of all ensemble members using a MLP network and obtain a prediction which is stable. RBF networks are trained based on different subsets of the whole network traffic.

In the field of intrusion detection, the big challenge is the problem of large datasets produced from network traffic. In our proposed method, we use fuzzy clustering method to divide the original dataset into smaller subsets so that the base classifiers can be trained easier and faster. We also create a good diversity in the ensemble members by using different subsets which have no intersection with each other. The outputs of different base classifiers are aggregated with a stacking module which learns the errors of different base models. Our goal is to also maximize the detection accuracy for classes with smaller number of examples e.g. U2R and R2L

Table 1. Ensemble methods proposed for Intrusion Detection

Authors	Architecture	Base Classifiers	Diversity Creation Method	Combination Method	Detection Type
Chebrolu et al. (2005)	Hybrid Ensemble	BN, CART	Feature selection using Markov Blanket Model and Decision Tree Analysis	Weighting and Winner-Takes-All	Misuse
Cordella et al. (2007)	Multi-Stage Classifiers	LVQ	Classifier-Specific Feature Selection	Cascade Classification	Misuse
Kim et al. (2014)	Hierarchical classifiers	C4.5, SVM	–	Two-stage classification	Misuse, Anomaly
Giacinto et al. (2003)	Modular Multiple Classifier System	MLP ANN	Service-Specific Feature Extraction	Majority vote, Averaging, Decision template, Naïve Bayes	Misuse
Giacinto et al. (2008)	Modular Multiple Classifier System	Parzen Density Estimation, v-SVC, K-means	Service-Specific Feature Extraction	Min,Max,Mean, Product	Anomaly
Govindarajan & Chandrasekaran (2011)	Hybrid Ensemble	MLP and RBF ANNs	Bagging	Majority vote	Anomaly
Mukkamala et al. (2005)	Linear Combination	ANN, SVM, MARS	different classifiers	Weighting and Majority Vote	Misuse
Peddabachigari et al. (2011)	Hybrid Ensemble	SVM, DT, SVM-DT	different classifiers	Weighting, Winner-Takes-All	Misuse
Abraham & Jain (2005)	Hybrid Ensemble	DT, LGP, Fuzzy Rule Set	different classifiers	not mentioned	Misuse

3 Ensemble Framework

In this section we describe our ensemble method in detail. First we explain the stacked generalization method for combining base classifiers. Then the ensemble framework is presented. After that we elaborate on its different modules.

3.1 Stacked Generalization

The idea behind stacked generalization (also called stacking) is to create a mapped example for each example in the original dataset [17, 36]. It uses the predicted labels of base classifiers as new features for the newly created examples. The target labels remain the same as the original dataset. The original examples of a dataset are first classified by each of the base classifiers. Then, the outputs of the base classifiers are used to make features in another space. The new features and the actual labels of the examples from the original dataset are put together to form a new dataset. Next, a *meta-learner* is trained by the new dataset. This meta-learner, which can be any kind of classifier, is able to reduce the errors made by each of the base classifiers through its training process. Therefore, the generalization ability of the system is improved.

3.2 The Proposed Ensemble System

Our ensemble system consists of three major modules: Fuzzy clustering, base classifiers using RBF networks, and stacking module. To make the ensemble, we first divide the original training dataset into several subsets using fuzzy clustering technique. Then we train the equal number of RBF networks using the created subsets. The *membership grades* produced by fuzzy clustering technique are used as special weights for the individual classifiers. Using these weights the predictions of all individual RBF classifiers are combined and aggregated by a MLP network which acts as a stacking combiner. We use a MLP network for the combination module because the size of the dataset given to the meta-learner is very large (the whole dataset). With such a large dataset the RBF networks use a tremendous amount of system resources to build the model. MLP networks use less memory and CPU than RBF networks in experimental environment when the number of examples is very high. The final classification is determined by the MLP network output. The ensemble framework is depicted in Figure 1 .

Our ensemble framework is established in two phases: training phase and testing phase. We separate the train and test datasets before going further. In

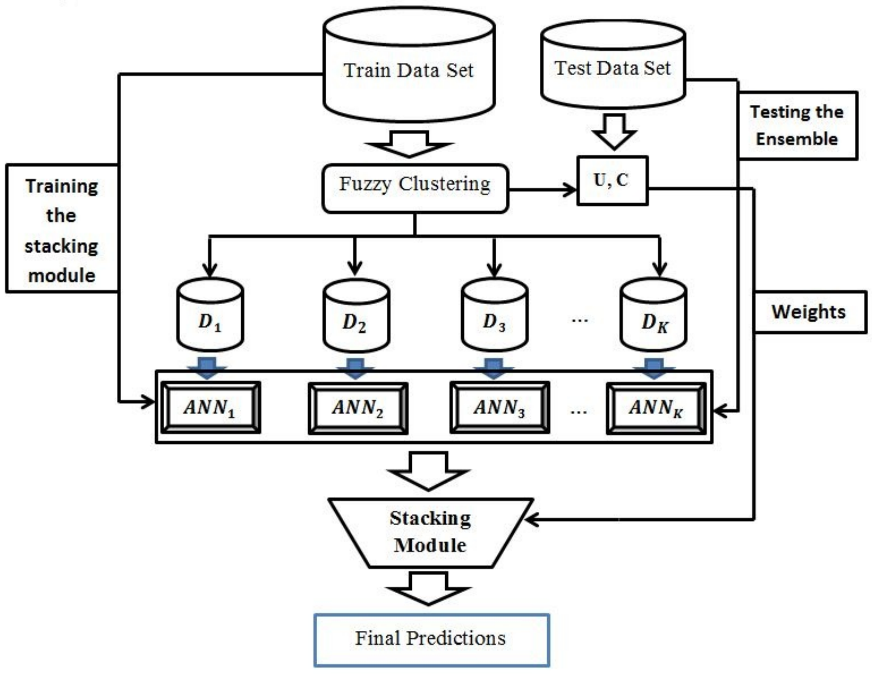

Figure 1. Framework of the ensemble classifier for IDS

the training phase the following tasks are performed:

- Task 1: From the training dataset D, K different subsets are created using fuzzy clustering technique.
- Task 2: For each training subset D_j, $j \in \{1, 2, \ldots, K\}$, ANN_j is trained using the subset so that it offers the best generalization error . This way, K different ANN base classifiers are created.
- Task 3: In order to reduce the error for every ANN_j, all base ANNs are simulated by the whole training dataset D and results are obtained. (The whole dataset D is given to all of ANNs as input and their predictions for the input examples are obtained as output.) To perform the stacking combination method the membership grades which are produced by the fuzzy clustering module are used to combine the results. Subsequently, we train a MLP network using the combined results.

In the testing phase, first, the membership grades of each test example are calculated based on the outputs matrices of fuzzy clustering module. Then, we simply give the testing examples to each of K RBF network base classifiers. After the classes of the examples are predicted by the base classifiers, the final results are computed using the stacking module.

In the next sections we explain how we deal with the problems in each module of the ensemble. We first describe subdividing the dataset using fuzzy clustering and then explain the training method for RBF network base classifiers. Finally, we describe how to combine the base models using the stacking combination method.

3.3 Fuzzy Clustering Module

This module aims to partition the training dataset into smaller subsets. We create diversity in the ensemble members through clustering the training data. Each subset includes the examples with similar characteristics and this way a group of similar examples are provided to an ANN for training. Since the size and complexity of the training subsets are reduced, the training of the base classifiers will be efficient, quick, and less complex.

We chose the popular Fuzzy C-means algorithm [37, 38] for fuzzy clustering module. In this method, each data point can belong to multiple clusters at the same time. But the degree of membership is determined by membership grades which are assigned to each data point. For each x_i in dataset D the fuzzy C-means algorithm assigns membership grade u_{ij} which shows the degree of x_i membership in cluster j $(0 \leq u_{ij} \leq 1)$. The membership grades are calculated for each example based on the minimization of an objective function which measures the distance between each data point and the cluster centers. If m is the size of the input dataset and K is the number of clusters, this objective function is calculated as follows:

$$J = \sum_{j=1}^{K} \sum_{i=1}^{m} u_{ij}^{q} \|x_i - c_j\|, 1 < q < \infty \qquad (1)$$

In the above equation q is the fuzziness exponent and can be any real value greater than 1 depending on the kind of problem. c_j is the center of j-*th* cluster and its dimensions are equal to that of input vector x_i.Creating the clusters is done through an iterative optimization process for objective function in which membership grades u_{ij} and cluster centers c_j are updated as below:

$$u_{ij} = \frac{1}{\sum_{d=1}^{K}\left(\frac{\|x_i - c_j\|}{\|x_i - c_d\|}\right)^{\frac{2}{q-1}}} \quad (2)$$

$$c_j = \frac{\sum_{i=1}^{m} u_{ij} x_i}{\sum_{i=1}^{m} u_{ij}} \quad (3)$$

Iteration is stopped when

$$\max_{i,j}\{|u_{ij}^{(t+1)} - u_{ij}^{t}|\} < \epsilon, (0 < \epsilon < 1) \quad (4)$$

where t shows the iteration step.

When Fuzzy C-means algorithm receives the unlabeled dataset with size m as input, it perform the above procedure and produces two matrices as output: Matrix U which has the membership grades of each data example in each of the K clusters and matrix C which includes the cluster centers for K clusters.

$$U = [u_{ij}], i = 1, 2, \ldots, m; j = 1, 2, \ldots, K \quad (5)$$

$$C = [c_j], j = 1, 2, \ldots, K \quad (6)$$

Now we create K disjoint subsets from the dataset using matrix U. To do this, we determine one subset for each individual example in the training dataset based on its maximum membership grade:

$$for\ each\ x_i : if\ u_{iw} = \max\{u_{ij}\} then\ x_i \in D_w, \quad (7)$$

where $i = 1, 2, \ldots, m; j = 1, 2, \ldots, K$.

After calculating the subset for all examples, the training dataset is divided to K disjoint subsets D_1, D_2,. . . ,D_K. The next step is to train K RBF neural networks using these K subsets.

3.4 RBF Neural Network Base Classifiers

We use RBF neural networks to learn the pattern of the subsets which we created in the last section. Each subset D_j is assigned to an RBF neural network and will be used to train the network. After training the networks, K base classifiers have been generated. We explain these classifiers in more details.

Two important types of artificial neural networks are multi-layer perceptron (MLP) and radial basis function (RBF) networks [11, 39]. MLP networks have one input layer, one output layer, and one or more hidden layers [40]. All nodes in the hidden and output layers have their own activation function. MLP networks usually are trained with back propagation (BP) algorithm. The BP algorithm is used to compute the weights between the input layer and the first hidden layer, between hidden layers and between hidden and output layers. This algorithm usually uses the gradient decent technique to adjust the weights.

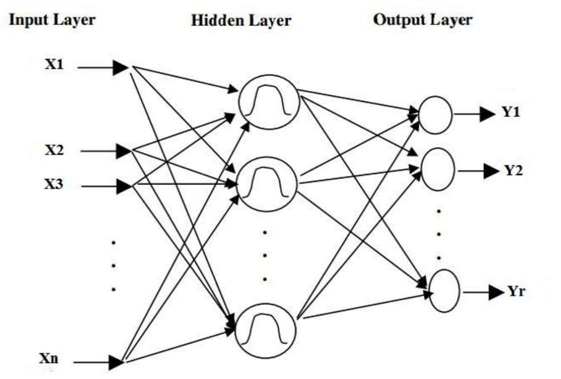

Figure 2. RBF network structure

The RBF neural networks consist of three constant layers. The input layer, one hidden layer, and the output layer [41].The neurons are completely connected but the difference here is that just the weights between the hidden layer and output layer are adjusted by training. Figure 2 shows the structure of a RBF network. The activations of hidden layer neurons are computed using radial basis function. The most widely used form of radial basis function is the Gaussian Kernel function which is calculated as follows:

$$g_i(\mathbf{x}) = \exp\left(\frac{-\|\mathbf{x} - v_i\|^2}{2\sigma_i{}^2}\right) \quad (8)$$

where $\mathbf{x}$ is the input vector and v_i is the vector denoting the center of the receptive field unit (hidden layer neuron) g_i with σ_i as its width parameter. This way, if we have s neurons in the hidden layer and r neurons in the output layer, the typical output of a RBF network will be calculated by:

$$y_j(\mathbf{x}) = \sum_{i=1}^{s} w_{ij} g_i(\mathbf{x}), \quad j = 1, 2, \ldots, r \quad (9)$$

where w_{ij} is the connection weight between the i-*th* receptive field unit and the j-*th* output, and g_i is the i-*th* receptive field unit.

The method of training RBF networks is a hybrid approach with two phases: unsupervised learning and supervised learning. First an unsupervised clustering algorithm is utilized to obtain the parameters of radial basis functions i.e. widths and the centers. Next, a supervised algorithm using least mean square error is performed to compute the weights of the connections between hidden nodes and the output nodes.

The reason why we chose to use RBF networks for this study is the advantages which this type of network has over the MLP type. For classification tasks, MLP has some drawbacks such as stopping at local minima and slow convergence. Moreover, before training the model we have to determine the number of hidden layers and hidden nodes, possibly based on the input dataset, which causes inflexible training. On the other hand, RBF network is capable of universal approximation using a single hidden layer, which removes the need for specifying the number of hidden layers and nodes. Furthermore, the simple linear transformation in the output layer can be optimized so that the learning algorithm becomes very fast and less likely to converge in the local minima [42]. The RBF learning is faster because it needs less computation.

We train K RBF networks with the K subsets from the train dataset to build the base models. These base models are different from each other because they have been trained on different subsets with no intersections. This makes a great diversity among ensemble members which is necessary for an ensemble system to work with high performance. After training the networks, K base models for the ensemble are ready to be combined using staking method.

3.5 Stacking Module

The proposed stacking module combines the predictions of the base classifiers using an efficient method and uses the result as a new set of features for another classifier. The new classifier will learn from the predictions of the base classifiers and gives final predictions. So the stacking module has two components: The combination unit and the MLP classifier.

3.5.1 Combination Unit

Each RBF network gives a prediction for the examples from the dataset. Then, these predictions will be combined to form the input of the meta-learner. The matrix U, which was produced by the fuzzy C-means technique, has the membership grades of the examples. These grades are used for assigning weights to the base classifiers. We follow the procedure bellow to combine the results:

(1) The whole training dataset D is given to all ANN_j base models and the predictions are obtained. The prediction for the example x_i is as follows:

$$P^i = [p_1^i, p_2^i, \dots, p_K^i], i = 1, 2, \dots, m \qquad (10)$$

where m is the size of the training set. p_j^i is the prediction vector for the *i-th* example by ANN_j.

(2) Multiply the membership grades of each example related to each subset to the corresponding predictions of the ANNs for the example. The new weighted prediction for example i will be:

$$O^i = [p_1^i u_{i1}, p_2^i u_{i2}, \dots, p_K^i u_{iK}] \qquad (11)$$

where u_{ij} is the membership grade of *i-th* example in the *j-th* cluster.

(3) We use a hybrid method to reduce the effect of majority models with small value predictions against the minority models with large value predictions. Also we do not want to ignore the prediction vectors of the majority models with acceptable large values even if their values are smaller than the maximum value provided by one model. Our hybrid method is inspired by the combination method used in Tian's paper [43]. In this method we use a threshold value (δ) which is a real number between 0 and 1. For an example in the dataset, consider $p_j = (y_j^1, y_j^2, \dots, y_j^c)$ as the weighted prediction vector of *j-th* base classifier for the example ($j = 1, 2, \dots, K$), c is the number of classes in the dataset. We preprocess the weighted outputs as follows:

(a) If $\forall t \in \{1, 2, \dots, c\} and\, \forall j \in \{1, 2, \dots, K\}$, $y_j^t < \delta$ then p_j is held intact. Then the weighted prediction vectors of all base models are aggregated linearly and the result is obtained.

(b) If the condition in (a) is not satisfied, the vectors p_j are processed as follows: If $y_j^t < \delta$ then $y_j^t = 0$. Next, the linear aggregation of the p_j vectors is calculated and the result is obtained.

Using this combination method along with setting a threshold, we avoid ignoring the predictions of minority base models which have large prediction values for a class (probability of an example being in a particular class) against majority ones with smaller values for other classes. Therefore, we signify the effect of weighting the classifiers and receive higher accuracy in classification.

3.5.2 MLP Classifier

The results of the final aggregation in the combination unit are used as a new input for the MLP network. The aggregated predictions for the samples are c by 1 vectors which are used to indicate new features for the samples in another feature space. The new dataset using all these vectors and their corresponding labels in the original dataset is given to the MLP network for training. This MLP network learns the errors made by different base classifiers and increases the accuracy of the total classification. After training the MLP with the new dataset the classification model is built.

The procedure described above is used for training the ensemble when we use the training dataset D. For testing the ensemble model or applying it to any new traffic we do the following:

(1) For any example in test dataset T, we first calculate its membership grade in each cluster which was created by the fuzzy clustering module. We use the matrix of cluster centers C for calculating the membership grades. For a new x_i^T the

membership grade u_{ij}^T is calculated with the following formula:

$$u_{ij}^T = \frac{1}{\sum_{d=1}^{K}\left(\frac{\|x_i^T - c_j\|}{\|x_i^T - c_d\|}\right)^{\frac{2}{q-1}}} \qquad (12)$$

(2) Give the test dataset T to all base classifiers. Their prediction vectors and the membership grades are given to the stacking module which combines their outputs using the methods described in combination unit. Then the combined vectors are given to MLP classifier and the final results are obtained which are the predicted classes for the test data.

4 Experiments and Results

We carried out some experiments to evaluate the performance of the ensemble system. For the experiments, we used NSL-KDD dataset which has some benefits over the original KDD99 dataset. The system was implemented in Matlab R2012a environment on a PC with Core i5 CPU and 4 GB of RAM and a 64 bit windows operating system. In the next sections we describe the experiments in more detail.

4.1 Data preprocessing

Traditionally, most of the research works conducted on the field of intrusion detection utilized the popular KDD99 dataset which has a large size and includes many duplicate and redundant records. The large size of the dataset causes the classification task to be long and exhausting. The very high number of redundant records makes the classification task to be biased toward the frequent samples. The detection for smaller classes i.e. U2R and R2L which are more harmful attacks will not be efficient enough. Furthermore, the existence of these redundant records in the test set will also cause the evaluation results to be biased by the algorithms which have better detection rates on frequent samples [44].

In this study, we used the NSL-KDD dataset [45] which is derived from original KDD-99 and has eliminated some of its drawbacks. It has the following characteristics [44]:

- There is no redundant record in the training set, so the bias toward more frequent record will not happen during learning process.
- There is no duplicate record in the new test set, so the evaluation of the learners will not be biased by the methods which have higher detection accuracy for the frequent records.

The NSL-KDD dataset consists of train and test sets. Each data sample is related to a network connection and consists of 41 features related to the connection characteristics of the network traffics. The labels of the samples indicate the class of attack type or the normal state of the connection. The attacks in the dataset are categorized into four main classes: DoS, Probe, U2R, and R2L.The train data and the test data include 125973 and 22544 records respectively.

Table 2. Number and percentage of records in each class in training and testing datasets

Class	Training Set		Test Set	
Normal	6000	31.5%	9711	42.9%
DoS	6000	31.5%	7458	33.0%
PRB	6000	31.5%	2421	10.7%
U2R	52	0.3%	7533	2.3%
R2L	995	5.2%	2471	10.9%

Revathi and Malathi performed a detailed analysis on NSL-KDD using different classification methods [46]. Their analysis showed that NSL-KDD dataset is ideal for comparing different intrusion detection models. They used two feature sets (one with all 41features and one with a reduced 15 features). In their experiments which were carried out by WEKA data mining Tool, Random Forest showed the highest accuracy compared to other methods in both cases. Chauhan et al. also used NSL-KDD dataset to perform a comparative analysis of classification techniques for intrusion detection. Kim et al. used NSL-KDD for evaluating their proposed hybrid model [25]. However, few studies have used this dataset in their experiments.

In order to reduce the size of the training set in our experiment, we selected 19047 records from the train set, in which all records in classes U2R and R2L were selected, and 6000 records were selected randomly from each of the Normal, DoS and PRB attacks. Table 2 shows the detailed information about the two datasets and the number of records in each class. In order to use the dataset for classification we need to normalize the data features. Therefore, using a linear normalization method, we convert the values of all features to the range (0, 1) except for features 5 and 6 which have very large values. These features are logarithmically scaled and converted to the values (0, 9).

4.2 Performance Measures

In order to evaluate the results of the experiments, some performance evaluation criteria are used [47]:

- True Positive (TP): number of examples correctly classified as being a particular type of attack.
- False Positive (FP): number of examples incorrectly classified as being a particular type of attack
- True negative (TN): number of examples correctly classified as not being a particular type of attack
- False Negative (FN): number of examples incor-

rectly classified as not being a particular type of attack.

A popular measure which is usually used for assessing the performance of classification tasks is Accuracy which is calculated based on the above criteria as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{13}$$

However, in intrusion detection the size of the classes are very different from each other. The number of samples in U2R, R2L, and PRB is very low compared to the other two classes. Therefore, evaluating the performance of the system on the test data would be biased considering the above criteria. So the above measures are not satisfactory to evaluate the performance in a standard way [48]. This is why we use three different performance measures which are precision, recall and F-value. These criteria are not dependent on the size of the training and testing samples and can be really helpful in assessing the performance of the ensemble model. They are determined as follows:

$$Precision, P = \frac{TP}{(TP + FP)} \tag{14}$$

$$Recall, R = \frac{TP}{(TP + FN)} \tag{15}$$

$$F - Value = \frac{(1 + t^2)PR}{t^2(P + R)} \tag{16}$$

where t indicates the relative importance of precision versus recall and normally is set to 1.

4.3 Implementation

A common problem in clustering methods is how to determine the right number of clusters. One possible solution is to perform experiments with different number of clusters and find the number which gives us the best performance in the ensemble. However, this method is complicated and time-consuming. In addition, we need to choose the number of clusters before training the base classifiers. In this study,we used the idea of clustering quality evaluation [37, 49] in order to identify the optimum number of clusters automatically. In [35] the authors used this method to avoid subjectivity when choosing the initial number of clusters. We followed the same procedure, used four evaluation methods and integrated the results to make better decisions about the initial number of clusters. After this step, the obtained optimum number of clusters was 8.

Data samples with 41 features were given to the RBF networks as input vectors. RBF networks were implemented with limited number of hidden neurons which is set before the implementation. The important parameters for the RBF networks are Goal, Spread, and MN. Goal indicates the desired mean squared error for the network learning process. Spread shows the width of radial basis function and MN is the maximum number of hidden neurons. The optimal set of parameters for the RBF networks was obtained based on the best generalization error of the networks. In our experiment Goal = 0.002, Spread = 0.8, and MN = 50. The input layer of the RBF networks has 41 nodes and the output layer has 5 nodes. The output of a RBF network for each example is a vector with 5 values indicating the support degrees for the example belonging to each of the 5 classes (probability of being a member in each of the 5 classes). The input vector for the MLP network is also 5 by 1 which shows the aggregated predictions of the base models. The output vector has the same dimension as the input vector. Since this experiment is rather complex, the number of hidden neurons in the MLP network is determined as 10.

Table 3. Precision (%) of single classifiers compared to ANN ensemble on 5 classes

Classifier	Normal	DoS	PRB	U2R	R2L
C4.5	83.3	95.8	79.8	11.6	82.3
KNN	85.3	95.6	83.5	58.6	68.3
Naive Bayes	81.5	92.6	41.7	1.2	59.7
ANN-ensemble	87.6	96.2	83.3	77.4	81.4

4.4 Results and Discussion

We performed experiments 10 times by randomly selecting the samples with the sampling rules described in Section 4.1 and obtained the average results. The accuracy with the best number of clusters (8 clusters) is 89.1%. We compared the results with the other well-known classification methods, Decision Tree (C4.5), Naïve Bayes and K nearest neighbor (KNN). The comparative results for the classifiers are depicted in Tables 3 to 5. We performed the comparison based on the three measures introduced in Section 4.2 separated by each class.

As can be seen from the tables, the ensemble method has the highest precision compared to all classifiers except for KNN in PRB attack. The recall of ensemble method is higher than all methods in Normal class and has a close value to the best recall in other classes. The overall performance of the ensemble method is demonstrated by the F-value, where it has significant superiority over other classifiers in the low frequent attack classes U2R and PRB.

In order to evaluate the performance of our ensemble strategy, we also implemented two popular ensemble methods Bagging and Boosting [50] for the detection of intrusion and compared their results with our proposed system. In Bagging, the ensemble is made

Table 4. Recall (%) of single classifiers compared to ANN ensemble on 5 classes

Classifier	Normal	DoS	PRB	U2R	R2L
C4.5	86.7	74.2	66.4	95.5	96.8
KNN	94.0	79.1	72.2	91.5	92.8
Naive Bayes	88.4	32.3	88.7	6.5	77.9
ANN-ensemble	94.9	81.5	86.6	91.0	93.0

Table 5. F-value (%) of single classifiers compared to ANN ensemble on 5 classes

Classifier	Normal	DoS	PRB	U2R	R2L
C4.5	84.9	83.7	72.5	20.7	89
KNN	89.4	86.6	77.4	71.5	78.7
Naive Bayes	84.9	47.9	56.7	2.0	67.6
ANN-ensemble	91.1	88.2	84.9	83.6	86.8

Table 6. Precision (%) of two popular ensemble methods compared to ANN ensemble on 5 classes

Classifier	Normal	DoS	PRB	U2R	R2L
Bagging	82.9	96.0	74.7	45.1	85.7
Boosting	86.1	94.4	64.6	57.2	78.3
ANN-ensemble	87.6	96.2	83.3	77.4	81.4

of classifiers built on bootstrap replicates of the training set. The classifier outputs are combined by the plurality vote. In Boosting, classifier ensemble is built incrementally, adding one classifier at a time. The classifier that joins the ensemble at each step is trained on a dataset selected from the original dataset and has the examples which are more difficult to classify than the examples in the last step. In our experiments, we used the popular algorithm *AdaBoost* [51] as a Boosting method. The comparative results are depicted on tables 6 to 8.

Bagging in R2L attack has better precision than our ensemble method (by 4.3%), however the ANN ensemble shows much better precision than bagging in classes U2R and PRB (differences are 32.3% and 8.6% respectively).The two ensemble classifiers have slightly better F-values only in classes DoS (Boosting with 89.3%) and R2L (Bagging with 91.1%) than the ensemble system. The F-value for our ANN ensemble is significantly higher in the two classes PRB and U2R.

If we consider the overall accuracy on all 5 classes, our ensemble system's accuracy has a better performance with 89.08% over all other classifiers. Figure 3 shows the overall accuracy of the classifiers.

4.4.1 The Effect of Clustering

The clustering module divides the large and heterogeneous training set into several smaller subsets which

Table 7. Recall (%) of two popular ensemble methods compared to ANN ensemble on 5 classes

Classifier	Normal	DoS	PRB	U2R	R2L
Bagging	94.6	75.3	74.7	97.5	97.3
Boosting	93.7	84.7	64.6	89.5	96.3
ANN-ensemble	94.9	81.5	86.6	91.0	93.0

Table 8. F-value(%) of two popular ensemble methods compared to ANN ensemble on 5 classes

Classifier	Normal	DoS	PRB	U2R	R2L
Bagging	88.4	84.4	77.7	61.7	91.1
Boosting	89.7	89.3	70.4	69.8	86.4
ANN-ensemble	91.1	88.2	84.9	83.6	86.8

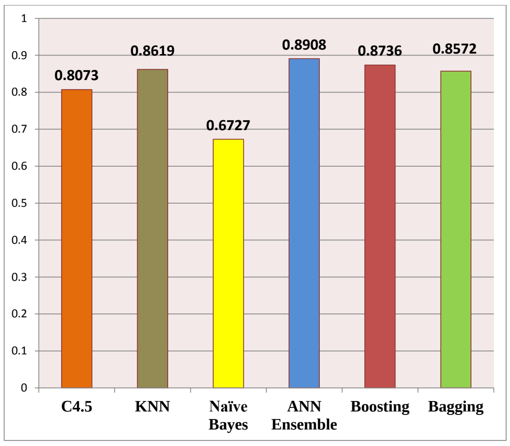

Figure 3. The overall accuracy of the classifiers

have less complexity and include more homogeneous samples. Therefore, clustering has an effect on the accuracy of the detection system by reducing the complexity of the base models and creating diversity in ensemble classifier. Figures 4, 5, 6 demonstrates the change in precision, recall and F-value of the ensemble system with different numbers of clusters in each of the five classes. As can be seen, the number of clusters can have impact on the detection, precision and recall for different classes. It is obvious that for large classes i.e. Normal and DoS the precision is relatively stable. For U2R and Probe attacks, the precision is the best from 4 to 8 clusters and decreases for more than 8 clusters. The precision of R2L class drops significantly when the cluster number increases and the reverse is true for its recall. The F-value for Probe, U2R and R2L attacks declines considerably when we increase the number of clusters from 8 to 10. It can be inferred that a particular number of clusters will optimize the detection performance of the ensemble system. However, choosing the right number of clusters before implementing the ensemble system is an

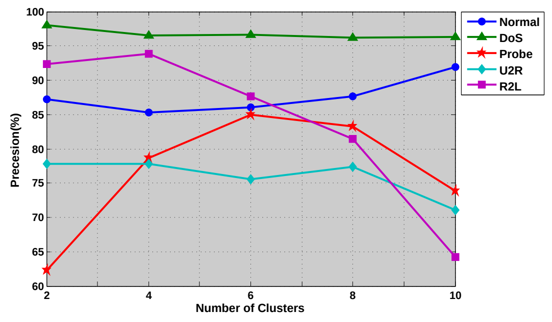

Figure 4. Precision in 5 classes with different number of clusters

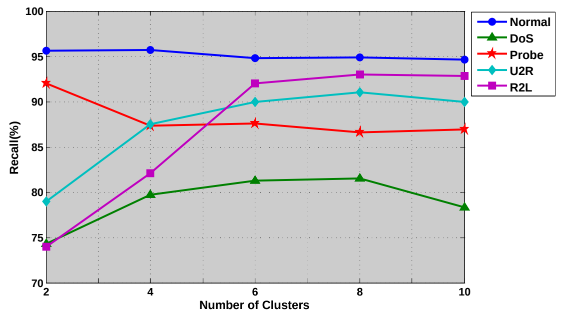

Figure 5. Recall in 5 classes with different number of clusters

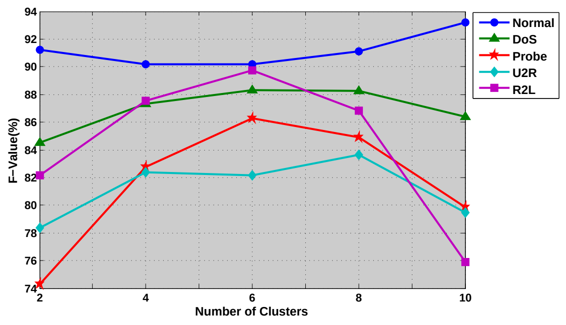

Figure 6. F-value in 5 classes with different number of clusters

issue which should be investigated further.

It is important to note that the values of the measures we provided here is not very high considering the values in other studies. However, this difference is justifiable considering the fact that we used NSL-KDD dataset and not the usual KDD99 dataset for the experiments. The KDD 99 dataset has many redundant records which makes bias in the predictions and raise the accuracy rate on some classes, especially the large classes. Therefore, the whole accuracy rate will increase. Furthermore, the accuracies of the other classification methods used in this study are also in the same range with our proposed system. This can prove that the system has comparable performance and the lower accuracy is due to the inherent characteristics of NSL-KDD dataset.

5 Conclusion and Future Work

In this article we proposed an ensemble classifier for intrusion detection based on fuzzy clustering and stacking combination method. Fuzzy clustering module divides the whole dataset into several homogeneous subsets. Therefore, the complexity of training subsets is reduced and the models can have more accurate classifications. The RBF neural networks are able to learn faster and easier than MLP networks and avoid stopping in the local minima. Using a hybrid combination method and a stacking procedure, the detection performance of the ensemble model is increased. The experiment with NSL-KDD dataset demonstrated the efficiency of our model particularly on smaller classes i.e. U2R, R2L and PRB which are difficult for classification. The stacking combination for ensemble learning has many areas for further exploration.

For future research, finding good strategies to combine the predictions with weights and aggregating the base models can be pursued. Another important area to work on is establishing a good method to weight base classifiers based on the intrinsic characteristics of each cluster. This way the accuracy of the ensemble model can further be improved.

References

[1] Carlos A Catania and Carlos García Garino. Automatic network intrusion detection: Current techniques and open issues. *Computers & Electrical Engineering*, 38(5):1062–1072, 2012.

[2] Aleksandar Lazarevic, Vipin Kumar, and Jaideep Srivastava. Intrusion detection: A survey. In *Managing Cyber Threats*, pages 19–78. Springer, 2005.

[3] Chih-Fong Tsai, Yu-Feng Hsu, Chia-Ying Lin, and Wei-Yang Lin. Intrusion detection by machine learning: A review. *Expert Systems with Applications*, 36(10):11994–12000, 2009.

[4] Nong Ye, Xiangyang Li, and Syed Masum Emran. Decision tree for signature recognition and state classification. In *Proceedings of IEEE Systems, Man and Cybernetics Information Assurance & Security Workshop*, 2000.

[5] Zonghua Zhang and Hong Shen. Application of online-training svms for real-time intrusion detection with different considerations. *Computer Communications*, 28(12):1428–1442, 2005.

[6] Latifur Khan, Mamoun Awad, and Bhavani Thuraisingham. A new intrusion detection system using support vector machines and hierarchical clustering. *The VLDB Journal—The International Journal on Very Large Data Bases*, 16(4): 507–521, 2007.

[7] James Cannady. Artificial neural networks for misuse detection. In *National information sys-*

tems security conference, pages 368–81, 1998.

[8] Guisong Liu, Zhang Yi, and Shangming Yang. A hierarchical intrusion detection model based on the pca neural networks. *Neurocomputing*, 70(7): 1561–1568, 2007.

[9] Yang Li and Li Guo. An active learning based tcm-knn algorithm for supervised network intrusion detection. *computers & security*, 26(7):459–467, 2007.

[10] Ke Wang and Salvatore J Stolfo. Anomalous payload-based network intrusion detection. In *Recent Advances in Intrusion Detection*, pages 203–222. Springer, 2004.

[11] Simon Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 2nd edition, 1999. ISBN 9780780334946.

[12] Josef Kittler, Mohamad Hatef, Robert PW Duin, and Jiri Matas. On combining classifiers. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 20(3):226–239, 1998.

[13] Nikunj C Oza and Kagan Tumer. Classifier ensembles: Select real-world applications. *Information Fusion*, 9(1):4–20, 2008.

[14] Robi Polikar. Ensemble learning. In *Ensemble Machine Learning*, pages 1–34. Springer, 2012.

[15] Tin Kam Ho, Jonathan J. Hull, and Sargur N. Srihari. Decision combination in multiple classifier systems. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 16(1):66–75, 1994.

[16] Nikunj C Oza and Kagan Tumer. Input decimation ensembles: Decorrelation through dimensionality reduction. In *Multiple Classifier Systems*, pages 238–247. Springer, 2001.

[17] David H Wolpert. Stacked generalization. *Neural networks*, 5(2):241–259, 1992.

[18] Kagan Tumer and Nikunj C Oza. Input decimated ensembles. *Pattern Analysis & Applications*, 6(1):65–77, 2003.

[19] Anders Krogh, Jesper Vedelsby, et al. Neural network ensembles, cross validation, and active learning. *Advances in neural information processing systems*, pages 231–238, 1995.

[20] Himadri Chauhan, Vipin Kumar, Sumit Pundir, and Emmanuel S Pilli. A comparative study of classification techniques for intrusion detection. In *Computational and Business Intelligence (ISCBI), 2013 International Symposium on*, pages 40–43. IEEE, 2013.

[21] Srinivas Mukkamala, Andrew H Sung, and Ajith Abraham. Intrusion detection using an ensemble of intelligent paradigms. *Journal of network and computer applications*, 28(2):167–182, 2005.

[22] Giorgio Giacinto, Fabio Roli, and Luca Didaci. A modular multiple classifier system for the detection of intrusions in computer networks. In *Multiple Classifier Systems*, pages 346–355. Springer, 2003.

[23] Giorgio Giacinto, Roberto Perdisci, Mauro Del Rio, and Fabio Roli. Intrusion detection in computer networks by a modular ensemble of one-class classifiers. *Information Fusion*, 9(1): 69–82, 2008.

[24] Luigi Pietro Cordella, Alessandro Limongiello, and Carlo Sansone. Network intrusion detection by a multi-stage classification system. In *Multiple Classifier Systems*, pages 324–333. Springer, 2004.

[25] Gisung Kim, Seungmin Lee, and Sehun Kim. A novel hybrid intrusion detection method integrating anomaly detection with misuse detection. *Expert Systems with Applications*, 41(4):1690–1700, 2014.

[26] Srilatha Chebrolu, Ajith Abraham, and Johnson P Thomas. Feature deduction and ensemble design of intrusion detection systems. *Computers & Security*, 24(4):295–307, 2005.

[27] Sandhya Peddabachigari, Ajith Abraham, Crina Grosan, and Johnson Thomas. Modeling intrusion detection system using hybrid intelligent systems. *Journal of network and computer applications*, 30(1):114–132, 2007.

[28] Ajith Abraham and Ravi Jain. Soft computing models for network intrusion detection systems. In *Classification and clustering for knowledge discovery*, pages 191–207. Springer, 2005.

[29] Lars Kai Hansen and Peter Salamon. Neural network ensembles. *IEEE transactions on pattern analysis and machine intelligence*, 12(10):993–1001, 1990.

[30] Giorgio Giacinto and Fabio Roli. Design of effective neural network ensembles for image classification purposes. *Image and Vision Computing*, 19(9):699–707, 2001.

[31] Zhi-Hua Zhou, Yuan Jiang, Yu-Bin Yang, and Shi-Fu Chen. Lung cancer cell identification based on artificial neural network ensembles. *Artificial Intelligence in Medicine*, 24(1):25–36, 2002.

[32] David West, Scott Dellana, and Jingxia Qian. Neural network ensemble strategies for financial decision applications. *Computers & operations research*, 32(10):2543–2559, 2005.

[33] Jianbo Yu. Online tool wear prediction in drilling operations using selective artificial neural network ensemble model. *Neural Computing and Applications*, 20(4):473–485, 2011.

[34] Srinivas Gutta, Ibrahim F Imam, and Harry Wechsler. Hand gesture recognition using ensembles of radial basis function (rbf) networks and decision trees. *International Journal of Pattern Recognition and Artificial Intelligence*, 11(06): 845–872, 1997.

[35] Ahmad Ghadiri and Nasser Ghadiri. An adaptive hybrid architecture for intrusion detection based

on fuzzy clustering and rbf neural networks. In *Communication Networks and Services Research Conference (CNSR), 2011 Ninth Annual*, pages 123–129. IEEE, 2011.

[36] Saso Džeroski and Bernard Ženko. Is combining classifiers with stacking better than selecting the best one? *Machine learning*, 54(3):255–273, 2004.

[37] James C Bezdek. *Pattern recognition with fuzzy objective function algorithms*. Kluwer Academic Publishers, 1981.

[38] Nikhil R Pal and James C Bezdek. On cluster validity for the fuzzy c-means model. *Fuzzy Systems, IEEE Transactions on*, 3(3):370–379, 1995.

[39] Gérard Dreyfus. *Neural networks: methodology and applications*. Springer, 2005.

[40] Lefteri H Tsoukalas and Robert E Uhrig. *Fuzzy and neural approaches in engineering*. John Wiley & Sons, Inc., 1996.

[41] Joydeep Ghosh and Arindam Nag. An overview of radial basis function networks. In *Radial basis function networks 2*, pages 1–36. Springer, 2001.

[42] Chunlin Zhang, Ju Jiang, and Mohamed Kamel. Intrusion detection using hierarchical neural networks. *Pattern Recognition Letters*, 26(6):779–791, 2005.

[43] Jin Tian, Minqiang Li, Fuzan Chen, and Jisong Kou. Coevolutionary learning of neural network ensemble for complex classification tasks. *Pattern Recognition*, 45(4):1373–1385, 2012.

[44] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali-A Ghorbani. A detailed analysis of the KDD CUP 99 data set. In *Proceedings of the Second IEEE Symposium on Computational Intelligence for Security and Defence Applications 2009*, 2009.

[45] Nsl-kdd dataset, 2009. URL http://nsl.cs.unb.ca/NSL-KDD/.

[46] S Revathi and A Malathi. A Detailed Analysis on NSL-KDD Dataset Using Various Machine Learning Techniques for Intrusion Detection. *International Journal of Engineering Research and Technology*, 2(12 (December-2013)), 2013.

[47] Stefan Axelsson. The base-rate fallacy and the difficulty of intrusion detection. *ACM Transactions on Information and System Security (TISSEC)*, 3(3):186–205, 2000.

[48] Paul Dokas, Levent Ertoz, Vipin Kumar, Aleksandar Lazarevic, Jaideep Srivastava, and Pang-Ning Tan. Data mining for network intrusion detection. In *Proc. NSF Workshop on Next Generation Data Mining*, pages 21–30, 2002.

[49] Weijiao Zhang, Chunhuang Liu, and Fangyu Li. Method of quality evaluation for clustering [j]. *Computer Engineering*, 20:004, 2005.

[50] Ludmila I Kuncheva. *Combining pattern classifiers: methods and algorithms*. John Wiley & Sons, 2004.

[51] Yoav Freund and Robert E Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational learning theory*, pages 23–37. Springer, 1995.

Mohammad Amini received his M.Sc. in Information Technology (E-commerce) from university of Qom in 2013. He is now a Ph.D. candidate studying E-commerce Engineering at Iran University of Science and Technology, Tehran, Iran. His main research interests include e-commerce security and applications, Data Mining, Machine Learning, and Decision Support Systems.

Jalal Rezaeenour received his M.Sc. and Ph.D. degrees in Industrial Engineering from Iran University of Science and Technology, Tehran in 2006 and 2011. His research interests are in Performance Measurement, Knowledge Management, Data Mining, Research Methods, Decision Making and Multivariate Data Analysis. He has published 2 books and has more than 65 papers in different journals and conferences. Dr. Rezaeenour is now the head of Department of Industrial Engineering, and director of Information and Communication Technology Administration in University of Qom.

Esmaeil Hadavandi is currently a Ph.D. Candidate in the Department of Industrial engineering and management systems, Amirkabir University of Technology, Iran. His research interests cover Soft Computing, Data Mining and Business Intelligence with a particular focus on developing decision support systems in areas such as strategic and operations management, Marketing and etc. Mr. Hadavandi is the author (and co-author) of many research publications in recognized international journals and conferences. He has also served as the reviewer of several international journals and conferences.