Journal of the Nigerian Association of Mathematical Physics
Volume **15 (November, 2009), pp 379 - 390**

A bagging approach to network intrusion detection

Adebayo O. Adetunmbi
Department of Computer Science
Federal University of Technology, Akure. Nigeria.

Abstract

Accompanying the benefits of Internet are various techniques of compromising the integrity and availability of the system connected to it due to flaws in its protocols and software widely entrenched. The presences of these flaws make a secured system a mirage for now, hence the need for intrusion detection system. In this paper, an ensemble approach – Bagging was used on five different machine learning techniques to improve accuracy of classifiers. Machine learning seeks for methods of extracting hidden pattern from data and come up with its own rules based on given data set. The five techniques were made up of two unsupervised (clustering) techniques – Kmeans and Fuzzy Rough C-means, and three supervised (classification) techniques – TreeReduct, LEM2 and Bayesian. Experimental study was carried out on the International Knowledge Discovery and Data Mining Tools Competition (KDD) dataset for benchmarking intrusion detection systems. The results generated from the experiment revealed that ensemble approach performance on the attack types and normal is slightly better or equal to the best performed algorithm on that particular class.

1.0 Introduction

The need for effective and efficient security on our system cannot be over-emphasized. This position is strengthened by the degree of human dependency on computer systems and the electronic superhighway (Internet) which grows in size and complexity on daily basis for business transactions, source of information or research. Intrusion detection system (IDS) is required to complement preventive security measures such as identification and authentication, logical access control, audit trails, encryption and decryption, digital signature and firewalls, to provide an additional layer of protection. Intrusion detection is meant to identify and detect unauthorized accesses or abnormal phenomena, actions and events in the system, which provides important information for timely countermeasures [3].

Basically, there are two approaches to intrusion detection model as described in [14]: Misuse detection model refers to detection of intrusions that follow well-defined intrusions patterns. It is very useful in detecting known attack patterns. Anomaly detection refers to detection performed by detecting changes in the patterns of utilization or bahaviour of the system. It can be used to detect known and novel attack. IDS are also classified as network-based or host-based in terms of source of data [15 and 36].

Majority of these IDSs are rule-based or expert system based. Their strengths depend on the ability of the expert that develops them. The massive deployments of these IDSs have

Telephone: +234-0803-961-7525

shown their operational limits and problems - false positive [8, 9, 13, 17 and 18]. Previous works of [8, 9, 11, 17, 24, 35 and 36] showed that there was need for development of a more effective and efficient intrusion based system.

The limitations of current intrusion detection systems led to an increasing interest in data mining and machine learning for intrusion detection. Early works on data mining approaches for intrusion detection includes the work of [12 and 40] but was first implemented in mining audit data for automated models for intrusion detection (MADAMID) [40]. Promising researches in this area include among others the work of [1, 2, 3, 5, 6, 15, 28, 32, 34, 37, 39, 40 and 41].

In this paper, three supervised learning techniques are used and two unsupervised learning techniques. TreeReduct [1] and LEM2 [20] are predictive algorithm based on the concept of Rough Set. Rough Set is a classical mathematical tool for feature extraction in a dataset which also generates explainable rules. Relevance features extracted are then used for classifying network traffic either as normal or attack. Naïve Bayes is a powerful tool for decision and reasoning under uncertain conditions; and it is based on strong independence assumption. Clustering is the process of grouping a set of physical or abstract objects into classes of similar objects.

This paper is organized as follows: in section 2 description of the intrusion detection evaluation dataset is presented followed by brief description of classification, clustering and ensemble technique employed in section 3. Section 4 presents the experimental setup and results followed by conclusion in section 5.

2.0 Intrusion data set

The data used in this paper are those proposed in the KDD'99 for intrusion detection which are generally used for benchmarking intrusion detection problems. The dataset was a collection of raw TCP dump data over a period of nine weeks simulated on a local area Network. The training data was processed to about five million connection records from seven weeks of network traffic and two weeks of testing data yielded around two million connection records. The training data is made up of 22 different attacks out of the 39 present in the test data. The attacks types are grouped into four categories: DOS, Probe, R2L and U2R since our focus is not to detect each attack type but the major category into which each fall. Table 2.1 shows the different attack types for both training (known) and the additional attack types included for testing (unknown) for the four categories. The four categories of the attacks described thus:

(1) DOS: Denial of service – e.g. syn flooding

(2) Probing: Surveillance and other probing, e.g. port scanning

(3) U2R: unauthorized access to local super user (root) privileges, e.g. buffer overflow attacks.

(4) R2L: unauthorized access from a remote machine, e.g. password guessing

The training dataset consisted of 494,021 records among which 97,277 (19.69%) were normal, 391,458 (79.24%) DOS, 4,107 (0.83%) Probe, 1,126 (0.23%) R2L and 52 (0.01%) U2R connections. The testing dataset is made up of 311,029 records out of which there were 60,593 (19.48%) normal, 229,853 (73.90%) DOS, 4,166 (1.34%) Probe, 16,189 (5.21%) R2L and 228 (0.07%) U2R. The test and training data are not from the same probability distribution. In each connection are 41 attributes describing different features of the connection (excluding the class attribute), and a label assigned to each either as an attack type or as normal.

Table 2.1: Known and novel attack types

DOS	Probe	R2L	U2R
Known			
Back, land, Neptune, Pod, smurf, teardrop	ipsweep, satan, nmap, portsweep	ftp_write, guess_passwd, warezmaster, warezclient, imap, phf, spy, multihop	rootkit, loadmodule, buffer_overflow, perl
Novel			
apache2, udpstorm, processtable, mailbomb	Saint, mscan	named, xlock, sendmail, xsnoop, worm, snmpgetattack, snmpguess	xterm, p.s., sqlattack, httptunnel

3.0 Basic concepts of clustering and classification approaches supervised (classification) techniques

3.1 Basic concept of rough set

Rough Set is a useful mathematical tool to deal with imprecise and insufficient knowledge, reduce data sets size, find hidden patterns and generate decision rules. Rough set theory contributes immensely to the concept of reducts. Reducts is the minimal subsets of attributes with most predictive outcome. Rough sets are very effective in removing redundant features from discrete data sets.

Rough set concept is based on a pair of conventional sets called lower and upper approximations. The lower approximation is a description of objects which are known in certainty to belong to the subject of interest, while upper approximation is a description of objects which possibly belong to the subset [23 and 31].

***Definition* 3.1:**

Let $S = \langle U, A, V, f \rangle$ be an information system, where U is a universe containing a finite set of N objects $\{x_1, x_2, ...x_N\}$. A is a non-empty finite set of attributes used in description of objects. V describes values of all attributes, that is, $V = \bigcup_{a \in A} V_a$ where V_a forms a set of values of the a^{th} attribute. $f: UxA \rightarrow V$ is the total decision function such that $f(x,a) \epsilon V_a$ for every $a \epsilon A$ and $x \epsilon U$. Information system is referred to as decision table (DT) if the attributes in S is divided into two disjoint sets called condition (C) and decision attributes (D) where $A = C \cup D$ and $C \cap D = \phi$.

$$DT = \langle U, C \cup D, V, f \rangle$$

A subset of attributes $B \subseteq A$ defines an equivalent relation (called Indiscernibility relation) on U, denoted as IND(B).

$$IND(B) = \{(x, y) \in UxU \mid f(x,b) = f(y,b) \forall b \in B\}.$$

The equivalent classes of B-indiscernibility relation are denoted $[x]_B$.

$$[x]_B = \{y \in U \mid (x, y) \in IND(B)\}$$

***Definition* 3.2**

Given $B \subseteq A$ *and* $X \subseteq U$. X can be approximated using only the information contained within B by constructing the B lower and B-upper approximations of set X defined as:

$$\underline{B}X = \{x \in X \mid [x]_B \subseteq X\}$$

$$\overline{B}X = \{x \in X \mid [x]_B \cap X \neq 0\}$$

***Definition* 3.3**

Given attributes $A = C \cup D$ and $C \cap D = \phi$. The positive region for a given set of condition attribute C in the relation to $IND(D)$, $POS_C(D)$ can be defined as

$$POS_C(D) = \bigcup_{x \in D^*} \underline{C}X$$

where D^* denotes the family of equivalence classes defined by the relation *IND(D).* $POS_C(D)$ contains all objects of U that can be classified correctly into the distinct classes defined by $IND(D)$.

Similarly, Given attributes subsets B, $Q \subseteq A$, the positive region contains all objects of U that can be classified to blocks of partition U/Q using attribute B. B is defined as:

$$POS_B(Q) = \bigcup_{x \in Q} \underline{B}X$$

***Definition* 3.4**

Given attributes B, $Q \subset A$, the degree of dependency of Q on B over U is defined as $\gamma_B(Q) = \dfrac{|POS_B(Q)|}{|U|}$

The degree of dependency of an attribute dictates its significance in rough set theory. Two rule induction techniques: Learning from Example Module version 2 (LEM2) and TreeReduct algorithms developed by [1 and 20] respectively are used in building an intrusion detection model.

3.2 The Bayesian classifier

In naïve Bayes classifier, instances to be classified are described by attribute vectors $\overset{\cup}{x} = (x_1, \Lambda\ , x_n)$. Bayes classifier assigns to instances most probable or maximum a posterior (MAP), classification from a finite set of c classes. Bayes classifier is given as:

$$c = \underset{c_j \in C}{\operatorname{argmax}} P(c_j) \prod_{i=1}^{n} P(x_i \mid c_j)$$

3.2.1 Unsupervised (Clustering) techniques

3.2.1.1 *K*-means clustering techniques

K-means (Hard *C*-Means (HCM)) is one of the simplest unsupervised learning algorithms for solving clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume *k* clusters) fixed a prior. The main idea is to define *k* centroids, one for each cluster. These centroids should be placed far away from each other as much as possible for better result, because different location causes different results.

The next step is to take each point belonging to a given data set and associate it to the nearest centroid. When no point is pending, the first step is completed and an early group age is done. At this point we need to re-calculate k new centroids, a new binding has to be done between the same data set points and the nearest new centroid. A loop has been generated. As a result of this loop, the k centroids change their location step by step until no more changes are done.

Finally, this algorithm aims at minimizing an objective function, in this case a squared error-function. The objective function $J = \sum_{j=1}^{k} \sum_{i=1}^{n} \| x_i^{(j)} - c_j \|^2$ where J is a chosen distance measure between a data point $x_i^{(j)}$ and the cluster centre c_j, is an indicator of the distance of the

n data points from their respective cluster centres.

K-means algorithm does not necessarily find the most optimal configuration, corresponding to the global objective function minimum. Also, the algorithm is significantly sensitive to the initial randomly selected cluster centres. The *k*-means algorithm can be run multiple times to reduce this effect. It is a good candidate for extension to work with fuzzy feature vectors.

3.2.1.2 **Fuzzy rough *C*-means**

Qinghua and Daren [33] proposed Fuzzy Rough clustering known as Fuzzy Rough C-means (FRCM), an improvement on rough k-means (RCM) proposed by [29] which was used for web users pattern mining. The principle behind FRCM is explained below.

According to the definition of lower and upper approximations in Rough Set theory, object set $[x_i]_B$ belong to the lower approximations if all the objects in $[x_i]_B$ are contained by X definitely. $[x_j]_B$ belong to upper approximation of *X* if the objects in $[x_J]_B$ are probably contained in object X based on knowledge *B*. Here knowledge B classifies the universe into three cases respect to certain object subset *X*: lower approximation, boundary region and negative region.

There are some elementary properties in rough set theory. Given an information system <U,A,V,*f*>, $B \subseteq A, U/B = \{X_1, X_2, ..., X_c\}$,:

Property 1 • $\forall x \in U, x \in \underline{B}X_i \Rightarrow x \notin \underline{B}X_j,\ j = 1,2,...,c;\ j \neq i$

Property 2 • $\forall x \in U, x \in \underline{B}X_i \Rightarrow x \in \overline{B}X_j,\ j = 1,2,...,c$

Property 3 • $\forall x \in U\ \forall_i, x \notin \underline{B}X_i \Rightarrow \exists X_k, X_l : x \in \overline{B}X_k\ and\ x \in \overline{B}X_l.$

Property 1 shows an object can be part of at most one lower approximation; property 2 shows that objects that belong to the lower approximation necessarily are contained by the upper approximations and the third property shows that if an object is not part of any lower approximation, the object must belong to at least two upper approximations.

HCM assigns a label to an object definitely; the membership value is 0 or 1. While Fuzzy C-means (FCM) maps a membership over the range 0 to 1; each object belongs to some or all of the clusters to some fuzzy-degree. For FRCM fuzziness memberships are imposed on the objects in the boundary due to the fact that lower approximations is the object subset which belongs to a cluster without doubt and boundary is the region assigned a label with uncertainty.

Here, the membership function is defined thus

$$u_{ij} = \begin{cases} 1, & x_k \in \underline{A}(v_i) \\ \dfrac{1}{\sum_{k=1}^{c} \left(\dfrac{\|x_i - c_j\|}{\|x_i - c_k\|} \right)^{\frac{2}{m-1}}}, & x_k \in \overline{A}(v_i) \quad i = 1,2,...c; k = 1,2,...,N \end{cases}$$

It is worth noting that the membership function is constructed and is not derived by the objective function. However, it was noted that this problem has little influence on the performance. Then the new centres are calculated by

$$c_i = \frac{\sum_{k=1}^{N} u_{ik}^m . x_k}{\sum_{k=1}^{N} u_{ik}^m}, i = 1,2,...,c.$$

The objective function used is $J_m = \sum_{k=1}^{N}\sum_{i=1}^{c} u_{ik}^{m} \left\| x_k - c_i \right\|^2$.

The criteria stated below is used to determine whether an object belongs to lower approximation or boundary region. For each object x and centre point c, $D(x,c)$ is the distance from x to c. The differences between $D(x,c_i)$ and $D(x,c_j)$ are used to determine the label of x.

Let $D(x,c_j) = \min_{1\le i\le c} D(x,c_i)$ and $T = \{\forall i, i \neq j : | D(x,c_i) - D(x,c_j) | \le Threshold\}$.

(1) If $T \neq \phi \Rightarrow x \in \overline{A}(c_i), x \in \overline{A}(c_j) and\ x \notin \underline{A}(c_l), l = 1,2,...,c$

(2) If $T = \phi, x \in \underline{A}(c_j), and\ x \in \overline{A}(c_j)$.

The definitions of lower and upper approximations are different from the classical ones. They are not defined based on any predefined indiscernible relation on the universe. In other word, FRCM first partitions the data into two classes: lower approximations and boundary. Only the objects in the boundary are fuzzified.

3.3 The ensemble classifier -bagging for intrusion detection

Ensemble classifier uses a combination of a set of models or classifiers, each of which solves the same original task in order to obtain a better composite global classifier with more accurate and reliable estimates or decisions than using a single classifier [7]. Minsky [26] opined that in solving really hard problems, several techniques should be combined in order to exploit the different virtues and evade the different limitations of each of these techniques. [25] reported that many experimental studies conducted by the machine learning community in recent years revealed that combining the output of multiple classifiers reduce the generalization error.

Ensembles approaches are very effective due to the fact that various types of classifiers have different "inductive biases" [19 and 27]. Also, this approach can effectively make use of diversity to reduce the variance-error without increasing the bias-error [7] and [38].

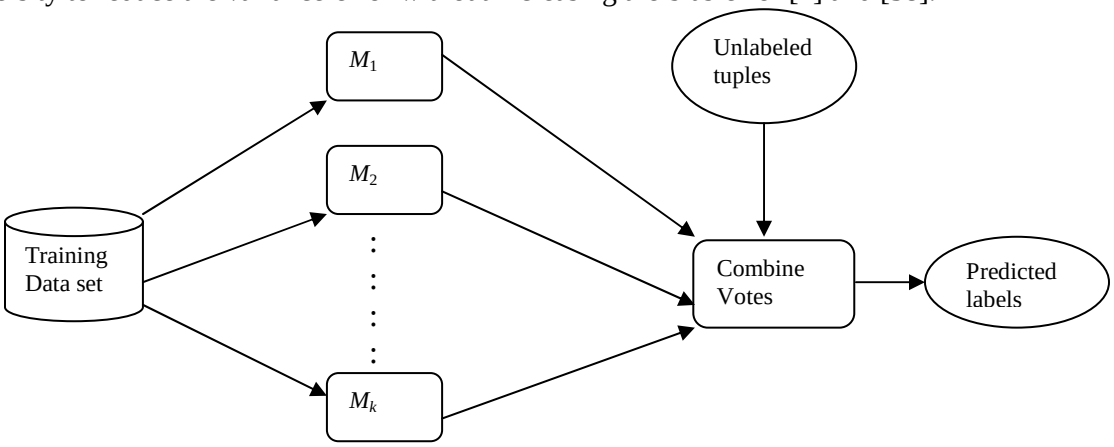

Figure 3.1: Multiple classifiers used in increasing model accuracy. Voting strategies are used to combine the prediction for a given unknown tuple (culled from [21]

Bagging and boosting are two techniques that could be used to improve the accuracy of classifiers. Other techniques could be found in [25]. Both bagging and boosting can be used for classification as well as prediction. Each combines a series of k learned models (classifiers or predictors), M_1, M_2,…, M_k, with the aim of creating an improved composite model, M^* as shown in figure 3.1.

3.3.1 Bagging

Bagging aims at improving the accuracy by creating an improved composite classifier, M* by amalgamating the various outputs of learned classifier into a single prediction. Given a set, *D*, of *d* tuples, bagging works as follows. For iteration *i* (*i* =1,2, …, *k*), a training set, D_i of *d* tuples is sampled with replacement from the original set of tuples *D.*

Since sampling with replacement is used, some of the original tuples of *D* may not be included in D_i, whereas others may occur more than once. A classifier model, M_i, is learned for each classifier M_i, returns its class prediction, which counts as one vote. The bagged classifier, *M** counts the votes and assigns the class with the most votes to *X*.

3.3.2 Bagging algorithm

Input:

D, a set of *d* training tuples
K, the number of models in the ensemble
A learning scheme (e.g. rough set, bayes, KCM, FRCM, etc)

Output:

A composite model M*

3.3.2.1 Method:

(1) for *i* = 1 to *k* do//create *k* models
(2) create bootstrap sample, D_i, by sampling *D* with replacement
(3) use D_i to derive a model M_i
(4) endfor

To use the composite model on a tuple, X
(1) if classification then
(2) let each of the k models classify *X* and return the majority vote
(3) if prediction then
(4) let each of *k* models predict a value for *X* and return the average predicted value;

Bagging algorithm creates an ensemble of models (classifier or predictors) for a learning scheme where each model gives an equally-weighted prediction.

4.0 Experimental setup and results

The KDD cup intrusion detection benchmark dataset earlier discussed in section two is used for the experimental purpose. Since the performance of all the machine learning techniques used for classification in this paper are at their utmost if used on discretized data set. Continuous variables are discretized using Shannon Entropy. Then, removals of redundant records in the training dataset were performed. The 145,738 records left in the training dataset were used in training the tree classification technbiques – LEM2, TreeReduct and Bayes.

The 22 attack types and normal in the training set were grouped into one of the five classes, 0 for Normal, 1 for probe, 2 for DoS, 3 for U2R and 4 for R2L. Preprocessing is grouped into three steps. In the first step, categorical features like protocol_type (3 different symbols tcp, udp,icmp), Service (66 different symbols), and flag (11 different symbols) were mapped to integer values ranging from 1 to *N* where *N* is the total number of symbol variation in each feature.

In the second step, continuous-value attributes like duration, src_bytes, dst_bytes are standardized based on equal bin partitioned into 20 for clustering and shanon entropy was used for classification techniques. Boolean feature like land having values 0 or 1 were left unchanged.

In the experiment attributes containing only one variation like attribute 20 and 21 (outbound command count for FTP session and hot login) identified to be of no significance in

Adetunmbi [3] were removed. Also attributes 13, 15, 17, 22 and 40 (Number of "compromised" conditions, root command attempted, Number of file creation operations, guest_login, and dst_host_rerror_rate) identified to be of little significant were also removed and a total of thirty four attributes left were used during testing. Table 4.1 shows the distribution of randomly selected data set for the experiment made up of one thousand, eight hundred and eight records.

Table 4.1: Distribution of the test Data

Categories	Number and names of attack	Total
Normal		500
Probe	Ipsweep(100), satan (100), nmap(100), portsweep(100)	400
DoS	Back(100), land(19), Neptune(100), Pod(100), smurf(133), teardrop (100)	552
U2R	rootkit(10), loadmodule(9), buffer_overflow(30), perl (3)	52
R2L	ftp_write(8), guess_passwd(53), warezmaster(20), Warezclient (200),imap(12),phf(4),spy(2), multihop(7)	304

4.1 Experimental results

For this experiment bagging approach is adopted. The approach was modified a little; the modifications include classification without replacement and assigning of different weight to different classifier based on the accuracy achieved for each group. The classifiers are TreeReduct, LEM2, EntropyBayes, *k*-means and FRCM. The first three are supervised based while the last two belongs to unsupervised learning techniques. The testing data set is the same as the one shown in Table 4.1

Tables 4.2, 4.3, 4.4, 4.5 and 4.6 show the confusion matrix obtained using the five different classifiers. The classification accuracy of each classifier for the four attack groups and normal is depicted in each confusion matrix.

Table 4.2: Confusion matrix obtained with *k*-means

Predicted as Actual	Normal	Probing	DOS	U2R	R2L
Normal(500)	**457(91.40%)**	0(0.00%)	42(8.40%)	1(0.20%)	0(0.00%)
Probing(400)	2(0.40%)	**186(46.50%)**	16(4.00%)	0(0.00)%	196(49.00%)
DOS(552)	180(32.61%)	1(0.18%)	**299(54.17%)**	0(0.00%)	70(12.68%)
U2R(52)	0(0.00%)	0(0.00%)	11(21.15%)	**25(48.08%)**	16(30.77%)
R2L(304)	4(1.32%)	0(0.00%)	69(22.70%)	1(0.33%)	**230(75.66%)**

Table 4.3: Confusion matrix obtained with FRCM ($\varepsilon = 0.25$)

Predicted as Actual	Normal	Probing	DOS	U2R	R2L
Normal(500)	**460(92.0%)**	0(0.00%)	39(7.8%)	1(0.20%)	0(0.00%)
Probing(400)	0(0.00%)	**198(49.50%)**	14(3.50%)	0(0.00)%	188(47.00%)
DOS(552)	180(32.61%)	1(0.18%)	**299(54.17%)**	0(0.00%)	70(12.68%)
U2R(52)	0(0.00%)	0(0.00%)	10(19.23%)	**30(57.69%)**	12(23.08%)
R2L(304)	4(1.32%)	0(0.00%)	64(21.05%)	1(0.33%)	**235(77.30%)**

Table 4.4: Confusion matrix obtained with EntropyBayes

Predicted as Actual	**Normal**	**Probing**	**DOS**	**U2R**	**R2L**
Normal(500)	**478(95.60%)**	2(0.40%)	2(0.40%)	4(0.80%)	4(0.80%)
Probing(400)	274(68.50%)	**125(31.25%)**	0(0.000%)	1(0.25)%	0(0.00%)
DOS(552)	96(17.39%)	2(0.36%)	**441(79.89%)**	13(2.36%)	0(0.00%)
U2R(52)	23(44.23%)	5(9.62%)	1(1.92%)	**23(44.23%)**	0(0.00%)
R2L(304)	212(69.74%)	12(3.94%)	0(0.00%)	23(7.57%)	**57(18.75%)**

Table 4.5: Confusion matrix obtained with TreeReduct

Predicted as Actual	**Normal**	**Probing**	**DOS**	**U2R**	**R2L**
Normal(500)	**336(67.20%)**	2(0.40%)	159(31.80%)	0(0.00%)	3(0.60%)
Probing(400)	0(0.00%)	**378(94.50%)**	0(0.000%)	0(0.00)%	22(5.50%)
DOS(552)	0(0.00%)	44(7.43%)	**492(89.13%)**	0(0.00%)	19(3.44%)
U2R(52)	0(0.00%)	0(0.00%)	0(0.00%)	**37(71.15%)**	15(28.85%)
R2L(304)	0(0.00%)	50(16.45%)	1(0.33%)	0(0.00%)	**253(83.22%)**

Table 4.6: Confusion matrix obtained with LEM2

Predicted as Actual	**Normal**	**Probing**	**DOS**	**U2R**	**R2L**
Normal(500)	**499(99.80%)**	1(0.20%)	0(0.00%)	0(0.00%)	0(0.00%)
Probing(400)	113(28.25%)	**287(71.75%)**	0(0.000%)	0(0.00)%	0(0.00%)
DOS(552)	360(65.22%)	0(0.00%)	**192(34.78%)**	0(0.00%)	0(0.00%)
U2R(52)	33(63.46%)	1(1.92%)	0(0.00%)	**18(34.62%)**	0(0.00%)
R2L(304)	180(59.21%)	0(0.00%)	0(0.00%)	2(0.66%)	**122(40.13%)**

The formula below is used in combining the five different classifiers involved in the ensemble approach to obtain the confusion matrix of table 4.7

$$BC_c = \sum_{c=i=1}^{5} W_{c,i}$$

where c – classifier,

W_i – Weight value for each class – normal and attacks

BC_c – summation of weights for each class/group – normal, and attacks

BC_c with the highest value gives new class group. In case of where two equal values are produced for BC_c, the class assigned is the one with the higher W_{ci} value.

Table 4.7: Confusion matrix obtained with bagging

Predicted as Actual	**Normal**	**Probing**	**DOS**	**U2R**	**R2L**
Normal(500)	**499(99.80%)**	1(0.20%)	0(0.00%)	0(0.00%)	0(0.00%)
Probing(400)	0(0.00%)	**384(96.0%)**	0(0.000%)	0(0.00)%	16(4.00%)
DOS(552)	32(0.06%)	0(0.00%)	**520(94.20%)**	0(0.00%)	0(0.00%)
U2R(52)	2(0.04%)	1(1.92%)	0(0.00%)	**37(83.22%)**	13(0.25%)

R2L(304)	1(0.003%)	39(0.13%)	11(0.04%)	1(0.003%)	**252(0.83%)**

Table 4.8 shows the correct classification for five different classifiers and Bagging for five different attack groups and normal using the standard intrusion detection evaluation dataset. LEM2 based rough set algorithm has the highest detection accuracy of 99.80% for normal while TreeReduct has the detection accuracy for all the attack groups as shown in Table 4.2. Finally, the result shows that the performance of the ensemble (bagging) approach is slightly better or equal to the best performed algorithm on that particular group.

Table 4.8: Performance of treereduct, LEM2, Roughbayes, *k* means, and FRCM in terms of Detection Accuracy

Class/Detector	TreeReduct	LEM2	EntropyBayes	Kmeans	FRCM	Bagging
Normal	**67.20**%	**99.80%**	95.60%	91.40%	92.00%	**99.80%**
Dos	**89.13%**	34.78%	79.89%	54.17%	54.17%	**94.20%**
Probe	**94.50%**	71.75%	31.25%	46.50%	49.50%	**96.20%**
R2l	**83.22%**	40.130%	18.75%	75.66%	77.30%	**83.22%**
U2r	**71.15%**	34.62%	44.23%	48.08%	57.69%	**71.15%**

5.0 Conclusion

In this paper, two clustering techniques (*k*-means and FRCM) were used to classify unlabelled dataset. From the experiment, the proposed technique FRCM performs better than *k*-means which shows that this is a promising approach. The detection accuracy of FRCM is better than of *k*-means as its performance outweighs kmeans in the detection of the presence of each group except DOS. Clustering algorithms are generally cheaper and of utmost importance for classifying unlabeled dataset. Also three predictive techniques were used and their performances in terms of detection accuracy are shown in Table 4.2. Finally, the ensemble approach performance on the attack types and normal is slightly better or equal to the best performed algorithm on that particular class.

The results of the developed tools are satisfactory though it can be improved upon. These tools will go a long way in alleviating the problems of security of data by detecting security breaches on computer system.

References

[1] Adetunmbi, A.O., Falaki, S.O., Adewale, O.S. and Alese, B.K. (2007a) A Rough Set Approach for Detecting known and novel Network Intrusion, Second International Conference on Application of Information and Communication Technologies to Teaching, Research and Administrations (AICTTRA, 2007), Ife, pp. 190 – 200.

[2] Adetunmbi, A.O., Alese, B.K., Ogundele, O.S. and Falaki, S.O. (2007b). A Data Mining Approach to Network Intrusion Detection, Journal of Computer Science & Its Applications, Vol. 14 No. 2. pp 24 -37.

[3] Adetunmbi, A.O. (2008) Intrusion Detection Based on Machine Learning Techniques, Ph.D. Thesis, Federal University of Technology Akure, Nigeria.

[4] Adetunmbi, A.O., Falaki, S.O., Adewale, O.S. and Alese, B.K. (2008) Intrusion Detection based on Rough Set and k-Nearest Neighbour, International Journal of Computing and ICT Research, vol. 2 No. 1. pp. 60-66. http:www.ijcir.org/volume1-number2/article7.pdf.

[5] Ajith, A., Ravi, J., Johnson, T., and Sang, Y.H. (2005). "D-SCIDS: Distributed soft computing intrusion detection system", Journal of Network and Computer Applications, Elsevier, pp. 1-19.

[6] Alan, B., Chandrika, P., Raheda, S., Boleslaw, S. and Mark, E.. (2002) Network-Based Intrusion detection using Neural Networks, www.cs.rpi.edu/~szymansk/paper/anie02.pdf

[7] Ali, K.M., and Pazzani, M.J. (1996). Error Reduction through Learning Multiple Descriptions, Machine Learning, vol. 24, no 3, pp. 173-202.

[8] Axelsson, S. (1999). The Base –rate Fallacy and Its Implication for the Difficulty of Intrusion Detection, In the proceeding of the 6th ACM Conference on Computer and Communication Security, pp. 127 -141.

[9] Axelsson, S.(2000a). Intrusion Detection Systems: A Survey and Taxonomy, Department of Computer Engineering, Chalmers University of Technology, Goteborg, Sweden. Technical Report TR-99-15. http://www.ce.chalmers.se/staff/sax/taxonomy.ps

[10] Axelsson, S. (2000b). Aspects of the Modelling and Performance of Intrusion Detection. Mphil thesis, Chalmers University of Technology, Sweden. http://www.ce.chalmers.se/staff/sax/

[11] Balasubramanuyan, J.S., Garcia-Fernadez, J.O., Isacoff, D., Spafford, E. and Zamboni, D. (1998). An Architecture for Intrusion detection using autonomous Agents, 14th Annual Computer Security Conference (ACSAC'98), pp. 13-24. IEEE Computer Society..

[12] Barbara, D. , Couto, J., Jajodia, R., Popyack, L., and Wu, N. (2001), "ADAM: Detecting Intrusions by Data Mining", Proc. IEEE Workshop on Information Assurance and Security, pp. 11-16.

[13] Benjamin, M., Ludovic, M., Herve, D. and Mireille, D. (2002). M2D2: A formal data Model for IDS alert correlation (RAID 2002), volume 2516 of Lecture Notes in Computer Science, pp. 115 – 137, Springer – Verlag.

[14] Biswanath, M., Todd L.H., And Karl, N.L. (1994) Network Intrusion Detection. IEEE Network, Vol. 8, no. 3, pp. 26-41.

[15] Byunghae, C., kyung, W.P., and Jaittyun, S. (2005). Neural Networks Techniques for Host Anomaly

[16] Intrusion Detection using Fixed Pattern Transformation in ICCSA. LNCS 3481, pp. 254-263.

[17] Cefriel (2003) Intrusion Detection Systems. http://www.cefriel.it/

[18] Eric, B., Bill, H., Alan, C., Clem, S., Lisa, T., and Jonathan, T. (2000), Data Mining for Improving Intrusion Detection, Technical Report. MITRE Corporation.

[19] German, S., Bienenstock, E. and Doursat, R. (1995) Neural Network and the bias/variance dilemma. Neural computation, 4: 1-58

[20] Grzymala-Busse, J.W. (1997) A new version of the rule induction system LERS. Fundam Inform. Vol. 3, pp. 27-39.

[21] Han, J. and Kamber, M. (2006) Data Mining Concepts and techniques, second edition, China Machine Press, pp. 296 -303.

[22] KDD Cup 1999 Data: http://kdd.ics.uci.edu/databases/kddcup99.

[23] Komorowski, J., Pokowski, L. And Skowron, A. (1998) Rough Sets: A Tutorial citeseer.ist.psu.edu/kom
orowski98rough.html

[24] Kumar, S.(1995) Classification and detection of computer intrusions. PhD thesis, Purdue University, USA.

[25] Maimon, O. and Rokach, L. (2005) Decomposition Methodology for Knowledge Discovery and Data Mining: Theory and Applications Series in Machine Perception and Artificial Intelligence, vol. 61 World Scientific series.

[26] Minsky, M. (1990) Logical vs Analogical or Symbolic vs. Connectionist or Neat vs. Scruffy, in Artificial Intelligence at MIT, Expanding Frontiers, Patrick , H. Winston (Ed.). Vol. 1. MIT Press.

[27] Mitchell, T. (1997) Machine Learning, McGraw-Hill.

[28] Mukkamala, S., Janoski, G., Sung, A. (2002). Intrusion detection using neural networks and support vector machines. In: Proceedings of IEEE Internation Joint Conference on Neural Networks, pp. 1702–1707.

[29] Pawlan, L.C.W. (2003). Interval set clustering of web users with rough k-means. International Journal of Intelligence Information System, pp. 5-16.

[30] Pawlak, Z. (1982) Rough Sets. International Journal of Computer and Information Sciences, Vol. 11, No. 5, pp. 341–356.

[31] Pawlak, Z. (1992) Rough Sets: Theoretical Aspects of Reasoning About Data. Kluwer Academic Publishing, Dordreiht.

[32] Pavel, L., Patrick, D., Christin, S. and Konrad, R. (2005). Learning intrusion detection: supervised or unsupervised?, International Conference on image analysis and processing, (ICAP), Italie, pp. 50-57.

[33] Qinghua, H. and Daren, Y. (2005) An Improved Clustering Algorithm for Information Granulation in L.Wang and Y. Jin (Eds): FSKD 2005, LNAI 3613, Springer Verlag Berlin Heidelberg, pp. 494 – 504.

[34] Sanjay, R., Gulati, V.P. and Arun, K.P. (2005) A Fast Host-Based Intrusion Detection System Using Rough Set Theory in Transactions on Rough Sets IV, LNCS 3700, pp. 144 – 161.

[35] Sodiya, A.S.(2004). A New Combined Strategy to Intrusion Detection, PhD Dissertation, University of Agriculture, Abeokuta, Nigeria.

[36] Sundaram, Aurobindo (1996). An Introduction to Intrusion detection ftp.cerias.purdue.edu/ pub/doc/intrusi
on_detection/Intrusion-Detection-Intro.ps.Z .

[37] Susan M. Bridges and Rayford B. Vaughnn (2000). Intrusion detection via fuzzy data mining, Proceedings of the 12th Annual Canadian Information Technology Security Symposium,Ottawa,Canada, June 19-23, pp.109-122.

[38] Tumer, K. and Ghosh, J. (1998)Robust Order Statistics based Ensembles for Distributed Data Mining. In Kargupta, H. and Chan, P., eds Advances in Distributed Knowledge Discovery, pp. 185 – 210, AAAI/MIT press.

[39[Wang, X. and He, F. (2006) Improving Intrusion Detection using Rough Set Theory and Association Rule Mining, International Conference on Hybrid Information Technology, (ICHIT '06) IEEE.

[40] Wenke, L. (1999). A Data Mining Framework for Constructing Features and Models for Intrusion Detection Systems. PhD dissertation, Columbia University, USA. http://www. Cc,gate
ch.edu/~wenke/

[41] Zhang Lian-hua, Zhang Guan-hua, YU Lang, Zhang Jie, and BAI Ying-cai (2004) Intrusion detection using rough set classification, Journal of Zhejiang University Science vol. 5, no. 9, pp. 1076-1086