ISSN 2278 - 3091

International Journal of Advanced Trends in Computer Science and Engineering (IJATCSE), Vol.2 , No.5, Pages :29-36 (2013)
Special Issue of ICCECT 2013 - Held during September 20, 2013, Bangalore, India

An Effective Combination Technique for Artificial intelligence based Ensembles for Intrusion detection

Krishan Kumar[1], Gulshan Kumar[2], Jabarweer Singh[3]
[1] Department of Computer Science and Engineering
Shaheed Bhagat Singh State Technical Campus, India, k.salujasbs@gmail.com
[2] Department of Computer Applications
Shaheed Bhagat Singh State Technical Campus, India, gulshanahuja@gmail.com
[3] Department of Computer Science and Engineering
Punjab Technical University, India, jabarweer@gmail.com

ABSTRACT

Artificial intelligence based techniques capitulate better performance in identifying the intrusions than other conventional approaches used in this field, but no single classifier is capable of identifying intrusions with acceptable accuracy. So, there is a necessity of integrating more than one classifier. This paper mainly focuses on constructing an ensemble of classifiers for classification of network traffic. In ensemble approach numerous machine learning algorithms are pooled. The main inspiration behind combining these classifiers is to take advantage of the powers of each classifier of the ensemble to get an overall more accurate classification. The existing combination techniques do not perform well when base classifiers are used with low accuracy or low diversity. The idea of the paper is to find promising classifiers for each type of network traffic class and combine them by proposing an effective combination technique so as to increase the overall accuracy.

This paper proposes a combination technique is based on weighted voting. It uses two weights, one weight is assigned to each traffic class and the second weight is assigned to each classifier for each class. Traffic classes are weighted in order of their error-rate of classification i.e. the class which is hard to classify is given more weight. Classifiers are weighted by their ability to classify the instance of the particular traffic class. The results indicate the superiority of the proposed combination technique over the other combination techniques. In all experiments, the ensemble was able to increase the overall accuracy over every individual classifier used.

Key words: Intrusion Detection System, ensemble, combination, Majority voting, artificial intelligence.

INTRODUCTION

The important security objectives include Availability, Integrity, Confidentiality, Accountability and Assurance [1]. These security objectives have to be satisfied so that we can protect crucial information, provide service to authorized users, authentication, access control and assured availability of resources. Various Security measures (tools and techniques) are used to assure these security objectives like cryptography, authentication, firewalls etc. Most of them work to prevent the malicious user attack what if the attack took place how to detect it and recover from it, for this purpose of detection and recovery from attack intrusion detection systems come into existence. An intrusion or attack can be defined as "any set of actions that attempt to compromise the security objectives" [2]. For classification of the network traffic in computer networks, intrusion detection System is used. Intrusion Detection is the method of inspection of the events taking place on a system or network and investigating them for intrusions, like non legitimate access, activity, or file manipulation. This includes three main courses of action: Monitoring and analyzing traffic; detection the intrusive activity; alarming the network administrator or taking some predefined actions [3]. Intrusion Detection System (IDS) is a tool that computerizes the intrusion detection procedure and sense potential intrusions. Intrusion Detection Systems provide three crucial safety measures: they examine, detect, and react to illegal activity by insiders and outsiders of the network [3]. A huge amount of activity data is collected from the network generating large log files and raw network traffic data, in which human inspection is impossible. Thus, these activity data are compressed into high-level events, called attributes. After it, a set of attributes is obtained and monitored by the IDS in order to detect intrusion attempts [2].

Various techniques are used in the process of detecting intrusion, the main types of techniques are statistical based techniques, knowledge based techniques and artificial intelligence based techniques. Ponce (2004) has listed several advantages of using AI based techniques over other approaches [4]. The major advantages include Flexibility; Adaptability; Pattern recognition; fast computing; high detection accuracy of new attacks. AIs can learn new rules automatically, whereas in traditional systems the security

administrator must add new rules for each new attack type or each new allowed program. Many researchers applied and evaluated AI-based techniques using different evaluation datasets for ID. They reported many challenges related to AI-based techniques and data set for ID. Some Researchers [5],[6],[7],[8] stated that single classifiers cannot generate the detection accuracy to an acceptance level hence there is a need of an approach having high detection accuracy which can be achieved by using multiple classifiers as they can complement each other's weaknesses. When we combine multiple classifiers then this architecture is known as an Ensemble of classifiers. Ensemble learning is applied to implement these multiple classifiers. Ensemble learning is a machine learning pattern where several learners are trained to resolve the same problem. As compared to conventional machine learning methods which aim to learn single hypothesis from training data, ensemble methods try to build a set of hypotheses and combine them [9].

In this paper first some AI based data mining techniques in intrusion detection are evaluated using the KDD99 dataset and on the basis of performance metric of these classifiers promising classifiers for each attack class are selected, then the output of these classifiers are combined by proposing a new combination technique. All experiments are done using Weka 3.6.8 and NSL-KDD data set.

BACKGROUND

The ensemble learning process has three phases: Ensemble generation; Ensemble selection; Ensemble integration. Ensemble generation is homogeneous if the same induction algorithm is used to generate all the classifiers of the ensemble, otherwise it is said to be heterogeneous. In ensemble generation phase, a pool of diverse base classifiers is generated. This can be done by using (1) different initialization parameters of base classifiers; or (2) different subsets of feature space (feature level); or (3) different data subsets (data level) to train the base classifiers. Ensemble selection requires selection of classifiers from pool of diverse base classifiers. Here different methods can be utilized to combine the pool of base classifiers obtained in ensemble generation phase: (1) combine all base classifiers; (2) combine smaller subsets according to clustering; (3) combine reduced sets of base classifiers that exceed specific thresholds of performance (i.e., overproduce and choose strategy). Ensemble integration involves the combination of predictions of a set of base classifiers selected in ensemble selection phase. It can use two different methods: (1) combination (also called fusion); or (2) selection [10]. The combination method consists in the combination of the predictions obtained by the different classifiers in the ensemble to obtain the final prediction. The selection approach, especially its dynamic form, selects one (or more) classifier from the ensemble according to the prediction performance of these classifiers on similar data from the validation set. The example of selection technique is MultiScheme in which a classifier is selected from among several using cross validation on the training data or the performance on the training data. Performance is measured based on the accuracy of the classifier. [11]

There are various methods used in combination like Majority voting method, Fuzzy theory method, Decision template method, Meta learning method etc. but majority voting is the most popular and widely used in combination because of its simplicity and capacity to deliver high accuracy when a large number of diverse classifiers are used. In simple Majority voting each classifiers act as a voter and the expected classification outputs are voting candidates. When an instance is encountered, based on the prediction of each classifier the vote is given to a particular class. The votes of all Classifiers are collected and the class with the highest number of votes is declared as the final prediction of the ensemble. In case of a tie arbitrary the prediction of any of the base classifiers is selected as final output. However, (unweighted) voting only makes sense if the learning schemes perform comparably well. If two of the three classifiers make predictions that are grossly incorrect, this could lead to degraded performance [12].

Hence it is better to assign weights to predictions i.e. assigning more weight to predictions that have more chances to be accurate this lead to weighted majority voting. Simple majority voting is a decision rule that selects one of the many alternatives, based on the predicted classes with the most votes. Weighted majority voting can be made if the decision of each classifier is multiplied by a weight to reflect the individual confidence of these decisions. Simple majority voting is a special case of weighted majority voting, assigning an equal weight of 1/k to each classifier where k is the number of classifiers in an ensemble [13].

RELATED WORK

Abraham et al. (2005) used the highest scored class as the final output among the base classifier outputs (Decision tree (DT), Support Vector Machine (SVM) and hybrid DT-SVM). According to the performance on the training data each classifier is assigned different weights. Using these weights and output of the classifier scores was calculated. Example, for class 1 if the DT works best, followed by the DTSVM and SVM model, the decision tree is assigned highest weight, followed by the hybrid decision tree-SVM model and SVM are assigned the lowest weight. For five different classes each classifier has different weights depending on their performance on the training data. So for a particular data record if all of them have different opinions, their scores are considered and the highest score one is declared the winner

and is the actual output of the ensemble approach.. This approach showed high detection rate for Probe (100%) and R2L (97.16%) type traffic and high improvement in detection of U2R (68%) than individual classifiers, but U2R detection rate is still low. There is a high computational load on the ensemble [8].

Chebrolu et al. (2004) first constructed the Bayesian network classifier and the CART models individually to obtain a very good generalization performance. They used the feature selection method using Markov blanket model and decision tree analysis. The ensemble approach was used for the 12, 17 and 41-variable data sets. They combined the classifier predictions in ensemble approach by using weighted voting as follows: each classifier output is given a weight (0-1 scale) depending on the generalization accuracy. If both classifiers agree then the output is decided accordingly. If there is a conflict then the decision given by the classifier with the highest weight is taken into account. The ensemble approach basically exploits the differences in misclassification (by individual models) and improves the overall performance [14].

Zainal et al. (2009) proposed an approach using Linear Genetic Programming (LGP), Adaptive Neural Fuzzy Inference System (ANFIS) and Random Forest (RF). The strengths from the individual models were evaluated and ensemble rule was formulated. First the hierarchical feature selection RST-BPSO (rough-discrete particle swarm optimization) for class-specific Feature Selection is applied and Ensemble classifier utilizes different learning mechanisms i.e., LGP, ANFIS, RF. Final ensemble prediction is the weighted voting of base classifiers. Detection accuracy was achieved by assigning proper weight to the individual classifiers in the ensemble. RF shows a better true positive rate for U2R class. Thus, by including the RF in the assemble model, the overall performance particularly the result for U2R class has improved [15].

Kim et al. (2011) used two weight vectors in weighted voting algorithm: a weight vector of classifiers and a weight vector of instances. The instance weight vector assigns higher weights to observations that are hard to classify. The weight vector of classifiers puts larger weights on classifiers that perform better on hard-to-classify instances. They proved that the iterated weight vectors converge to the optimal weights which can be directly calculated from the performance metrics of classifiers in an ensemble. The final prediction of the ensemble is obtained by voting using the optimal weight vector of classifiers. They verified that their method reduces the classification bias while maintaining a low variance. For the performance comparison between WAVE and other ensemble methods, they implemented an experiment using 28 real or simulation datasets. They demonstrated that the WAVE consistently outperforms Bagging that uses simple majority voting scheme. Although not significant, they observed a tendency that WAVE may perform better than Boosting. WAVE showed compatible performance with Random Forest. When they compared ensemble methods and pruned trees, WAVE showed the most consistent and significant improvement over the pruned trees [13].

TOOL AND DATSET

Tool: Weka (Waikato Environment for Knowledge Analysis)

Simulation work for proposed Network Intrusion Detection System is done using WEKA. Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes. [16]. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. WEKA consists of an Explorer, Experimenter, Knowledge flow, Simple Command Line Interface, Java interface.

Performance terms and metrics used in Weka:

Weka constructs a Confusion matrix (With c classes the confusion matrix becomes a $c \times c$ matrix containing the c correct classifications (the major diagonal entries) and $c2-c$ possible errors (the off-diagonal entries)) [17**]** which give us: TP: Number of True positives; FP: Number of False positives; TN: Number of True Negatives; FN: Number of False Negative. Various metrics are derived from confusion matrix to evaluate the classifiers performances these are: Accuracy is how accurately a classifiers classify instances; True Positive Rate (TPR) or Sensitivity or recall is the how well the classifier classify the positive instances; False Positive Rate (FPR) rate of negative instances are inaccurately classified by classifier; Precision is the probability of correctness of a positive prediction; F-measure c is the harmonic mean of precision and recall and can be used as a single measure of performance [17]. These performance metrics are calculated by following equations:

$$Accuracy = \frac{TP + TN}{n} \quad (1)$$

$$TPR = \frac{TP}{TP + FN} \quad (2)$$

ISSN 2278 - 3091

International Journal of Advanced Trends in Computer Science and Engineering (IJATCSE), Vol.2 , No.5, Pages :29-36 (2013)
Special Issue of ICCECT 2013 - Held during September 20, 2013, Bangalore, India

$$FPR = \frac{FP}{TN + FP} \quad (3)$$

$$precision = \frac{TP}{TP + FP} \quad (4)$$

$$F - measure = \frac{2 \times precision \times recall}{precision + recall} \quad (5)$$

Receiver Operating Characteristics (ROC) graphs have long been used in signal detection theory to depict the tradeoff between hit rates and false alarm rates over noisy channels

Dataset: NSL-KDD

Knowledge Discovery and Data Mining Competition - KDD Cup 99 [18] is the data set used for The Third International Knowledge Discovery and Data Mining Tools Competition, which was held in conjunction with KDD-99 the Fifth International Conference on Knowledge Discovery and Data Mining. The raw training data were about four gigabytes of compressed binary TCP dump data from seven weeks of network traffic. This was processed into about five million connection records. Similarly, the two weeks of test data yielded around two million connection records.

Table 1: Training data attack types

DoS	*Probe*	*R2L*	*U2R*
back	ipsweep	ftp_write	buffer_overflow
land	nmap	guess_passwd	loadmodule
neptune	portsweep	Imap	perl
pod	satan	multihop	rootkit
smurf		phf	
teardrop		spy	
		warezclient	
		warezmaster	

A connection is a sequence of TCP packets starting and ending at some well defined times, between which data flows to and from a source IP address to a target IP address under some well defined protocol. Each connection is labeled as either normal, or as an attack, with exactly one specific attack type. Each connection record consists of about 100 bytes [19].

In the KDD99 database, any network connection (or instance) is comprised of 41 attributes and each instance is labelled either as normal or as an attack-specified type. In KDD99 database, there are 4,898,430 labelled and 311,029 unlabeled connection records in the dataset. The labelled connection records consist of 22 different attack types categorized into 04 classes namely (DoS: denial of service, Probe, U2R: user to root, R2l: Remote to local) The 22 attacks types of labeled data are shown in Table 1 [20].

Whereas unlabeled dataset consists of 20 known and 17 unknown attacks these are shown in Table 2 [21].

Table 2: Test data attack types

DoS	*Probe*	*U2R*	*R2L*
Smurf	Ipsweep	Buffer_overflow	**Snmpgetattack**
Pod	**Saint**	Perl	**Named**
Apache2	Portsweep	**Xterm**	**Xlock**
Udpstorm	Satan	**Ps**	Multihop
Processstable	**Mscan**	Root kit	**Xsnoop**
Neptune	Nmap	Loadmodule	**Sendmail**
Back		**Httptunnel**	Guess_passwd
Mailbomb		**Sqlattack**	Phf
Teardrop			Warezmaster
Land			Imap
			Worm
			Ftp_write
			Snmpguess

In Table 2 the attack types with simple letters is known attacks and those written in bold letters are unknown attacks. The labelled dataset is used for training and unlabeled dataset is used for testing of the classifiers. The 17 unknown attacks used for testing helps in determining the accuracy of the classifiers for unknown attacks.

ISSN 2278 - 3091
International Journal of Advanced Trends in Computer Science and Engineering (IJATCSE), Vol.2 , No.5, Pages :29-36 (2013)
Special Issue of ICCECT 2013 - Held during September 20, 2013, Bangalore, India

It is clear that the total number of connection records to be used for training and testing of the classifiers is very large and also the number of connection records related to U2R and R2L is very less as compared to other attack classes. Moreover, KDD'99 is built based on the data captured in DARPA'98 which has been criticized by McHugh (2000), mainly because of the characteristics of the synthetic data. As a result, some of the existing problems in DARPA'98 remain in KDD'99 [22]. Due to shortcomings and large number of records on which test was very difficult to perform Tavallace et al. (2009) included 125,973 and 22,544 records in a training dataset and test dataset respectively by randomly selecting connections from KDD'99 dataset. They named this dataset NSL-KDD [23]. Further, in order to reduce non-uniformity in the dataset, we randomly selected maximum of 10,000 connection records of each attack type from labeling data set for the purpose of training the classifiers in an unbiased manner. Total 66,961 (including normal) connection records are selected from entire labeled KDD dataset for training of classifiers. In order to test the classifiers, we randomly selected 5000 connection records of all attack types from an unlabeled dataset. There are 40,603 connection records in the test dataset [21].

For using the dataset in training and testing purposes we must first preprocess the dataset. The preprocessing includes two phases: 1) Mapping of symbolic value features to numeric value; 2) Normalization of continuous features. We have done the preprocessing as suggested by Gulshan et al. (2010) as follow: 1) Symbolic features like protocol type, (3 different symbols), service (70 different symbols), and flag (11 different symbols) were mapped to integer values ranging from 0 to N-1 where N is the number of symbols; 2) The attack type feature is mapped to one of attack classes namely Probe, DoS, U2R and R2L; 3) The normalization of continuous features is done as per equation shown below:

$$Normalizedvalue_i = normalize(\ln(val_i + 1)) \quad (6)$$

$$normalize(X_i) = \frac{x_i - \ln(Min_i + 1)}{\ln(Max_i + 1) - \ln(Min_i + 1)} \quad (7)$$

THE PROPOSED COMBINATION TECHNIQUE

We have proposed a new weighting scheme for weighted voting which is based on the error rate of classification of data and precision of the classifier.

WVEP (Weighted voting based on error and precision):

The concept of weighted voting is to assign weights to the prediction of classifiers on the basis of their accuracy. Kim et al. [2013] suggested assigning weights to both the instance and classifiers [13]. So on the basis of this concept the proposed technique uses two weight vectors: a weight vector for the class type and second weight vector for the classifier. The first weight vector of the class type is calculated from the average error rate of the all classifiers for class I, we have calculated the average error of seventeen classifiers for all classes which is given in the table. The second weight vector which is to be assigned to classifier is calculated at the training time which on the basis of probability of accurate classification of the class by the classifier. Hence the weight to the prediction of a classifier is assigned by multiplying these two weights which is how hard it is to classify the class and how accurately the classifier classy that particular class. Average errors in the classification of 17 classifiers for five classes are:

Table 3: Average of classification error of 17 classifiers

Class Type	Average classification error
Wt (Normal)	4.14
Wt (Probe)	21.41
Wt (DoS)	24.35
Wt (R2L)	88.24
Wt (U2R)	83.96

And on the basis of following equation the final weight $F_t(x)$ for class i is determined by:

$$F_i(x) = \sum_{j=1}^{\max} W_i \times W_{ij} \quad (8)$$

Where,

W_i is the weight assigned to the class i on the basis of an error in its classification,

W_{ij} is the weight assigned to the classifier j if it classify the instance as i at training time on the basis of the precision of the classifier. The class I with the highest value is declared as the final output of the ensemble. In case of a tie the output of the classifier with higher detection accuracy is declared the final prediction of the ensemble.

The technique calculates the function $F_t(x)$ for each class i and the class with the highest value $F_t(x)$ is taken as the final prediction of the ensemble. In case of a tie we have taken the prediction as final of the classifier with the highest detection accuracy among the base classifiers used in ensemble. Our

ISSN 2278 - 3091
International Journal of Advanced Trends in Computer Science and Engineering (IJATCSE), Vol.2 , No.5, Pages :29-36 (2013)
Special Issue of ICCECT 2013 - Held during September 20, 2013, Bangalore, India

technique is another way of assigning weights to the classifiers in weighted voting. Hence it could be seen as an improvement to the weighted majority voting scheme which optimally assign weights.

Working of WVEP:

For combining predictions of an ensemble, Let there be "n" selected classifiers to combine and there are three classes a, class b, class c in the traffic to classify, WVEP works as follows:

1. For each classifier repeat steps 2 to 6.
2. Train the classifier.
3. Calculate precision for each traffic class.
4. Test the classifier on test data set.
5. Calculate the error rate of classifier for each class.
6. Calculate the detection accuracy of the classifier.
7. For each class calculate the average of error of classification for that class by all classifiers.
8. For each classifier j for an incoming instance repeat 9.
9. Do {Multiply the "average error rate of i" with "precision of j for class i" and assign this weight to the prediction of classifier j} While (classifier j predicts the instance as of class i)
10. Calculate the total weight for particular class prediction (let it be class i) i.e. add the weights of predictions that are same and assign it to the class i.
11. If (all classes don't have same weight) {Class with the highest weight is final prediction}; Else {the prediction of classifier with highest detection accuracy (calculated on test dataset) among the base classifiers is the final prediction}.

EXPERIMENTS AND RESULTS

The results are compared on KDD99 dataset of 6763 instances. . The new technique was able to improve the accuracy to 13.63% from majority voting and 18.39% from Multischeme. The proposed ensemble achieved detection accuracy of 82.05%. Figure 1 shows the accuracies of majority voting, Multischeme and the proposed combination technique and Figures from 2 to 6 shows the comparison of TPR and FPR of the proposed technique with Majority voting and Multischeme.

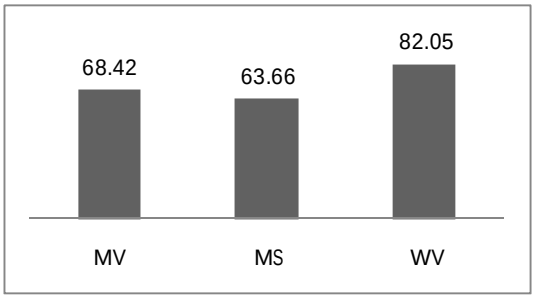

Figure 1: DA of WVEP vs. MV and MultiScheme

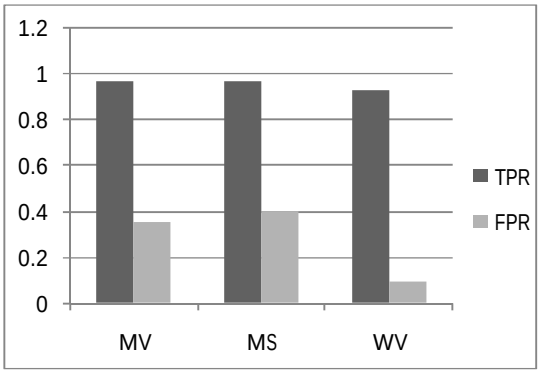

Figure 2: TPR and FPR for Normal Class

For normal class the technique was able to maintain the TPR near about existing techniques and the false positive rate was reduced by a large factor.

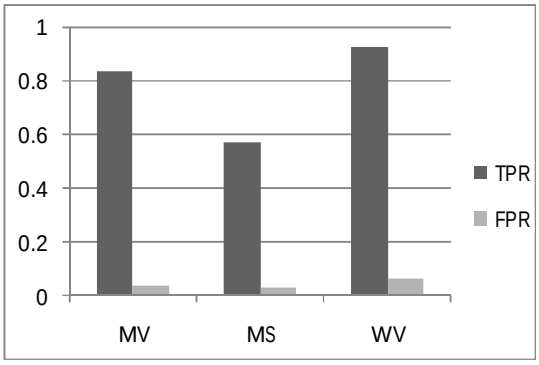

Figure 3: TPR and FPR for Probe Class

As we can see the majority vote and Multischeme is outperformed by proposed technique in case of Probe and DoS class types.

ISSN 2278 - 3091
International Journal of Advanced Trends in Computer Science and Engineering (IJATCSE), Vol.2 , No.5, Pages :29-36 (2013)
Special Issue of ICCECT 2013 - Held during September 20, 2013, Bangalore, India

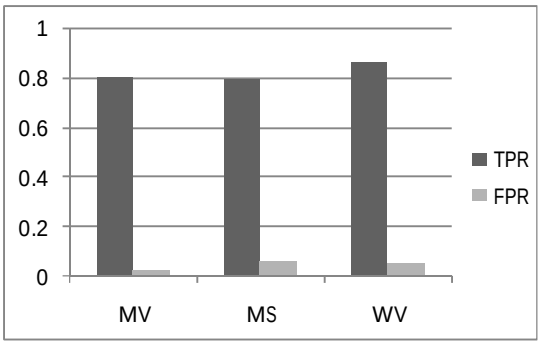

Figure 4: TPR and FPR for DoS Class

The True positive rate of the ensemble is increased using the proposed combination technique than using Majority voting or MultiScheme for combination.

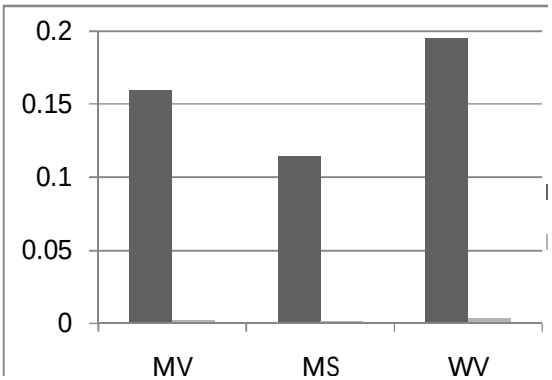

Figure 5: TPR and FPR for U2R Class

In every class except normal type the proposed technique was able to increase the true positive rate from other combination techniques but in case R2L class with acceptable detection accuracy and comparably.

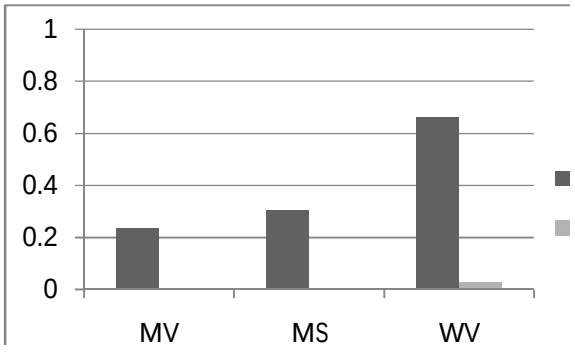

Figure 6: TPR and FPR for R2L Class

Some combination techniques are not able to increase detection accuracy when more than half of the base classifiers in the ensemble have low accuracy.

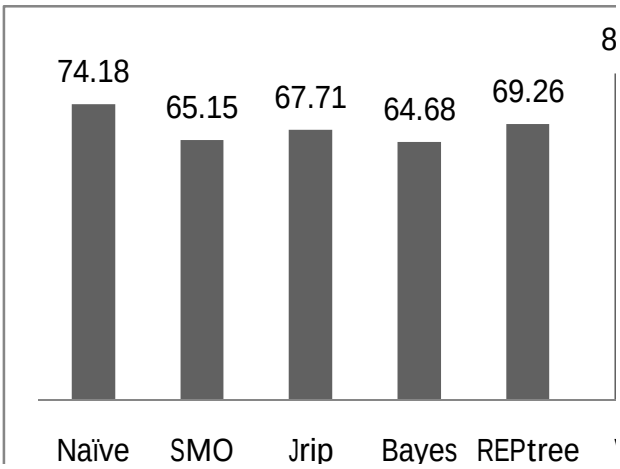

Figure 7: DA of proposed ensemble vs. base classifiers

The proposed scheme always achieves accuracy more than the classifiers used as base classifiers. As seen from the Figure 7, the proposed method has detection accuracy more than every individual classifier used in the ensemble.

CONCLUSION

The Simple Intrusion Detection System uses a single classifier for classifying data but many researchers proved that no single classifier can classify the traffic with high accuracy, so to achieve more performance, more than one classifier should be combined and Ensembles are used. The main idea is to exploit the strengths of each algorithm of the ensemble to obtain a robust classifier. We proposed a new weighted voting combination technique WVEP, for combining the predictions of multiple classifiers. We have shown that the proposed technique outperforms simple Majority voting and Multischeme combination approaches. We have also shown that the technique works well even if the classifiers are not diverse or highly accurate. The resulting accuracy was better than every single classifier used in combination.

In this work the proposed combination technique has been applied to a small number of classifiers selected from a limited pool of base classifiers but a better ensemble can be constructed by using a large number of classifiers. In future direction a large pool of diverse classifiers could be generated and based on some performance metric a set of classifiers can be selected from that pool and combined by using the proposed technique. The weight assigning function in the proposed technique may be optimized to assign voted weight more preciously by including some other measures. So, the future Scope may include the optimization of this combination technique and applying it to a large set of diverse classifiers.

REFERENCES

1. Stoneburner, Gary. **SP 800-33. Underlying Technical Models for Information Technology Security.** (2001).
2. G. Kumar, K. Kumar, and M. Sachdeva, **The use of artificial intelligence based techniques for intrusion detection: a review,** *Artificial Intelligence Review*, vol. 34, no. 4, pp. 369–387, 2010.
3. V. Marinova-Boncheva, **A short survey of intrusion detection systems,** *Problems of Engineering Cybernetics and Robotics*, vol. 58, pp. 23–30, 2007.
4. M. C. Ponce, **Intrusion detection system with artificial intelligence,** in *FIST Conference-June*, 2004.
5. T. G. Dietterich, **Ensemble methods in machine learning,** in *Multiple classifier systems*. Springer, 2000, pp. 1–15.
6. M. Sabhnani and G. Serpen, **Application of machine learning algorithms to kdd intrusion detection dataset within misuse detection context.** in *MLMTA*, 2003, pp. 209–215.
7. S. Mukkamala, A. H. Sung, and A. Abraham, **Intrusion detection using an ensemble of intelligent paradigms,** *Journal of network and computer applications*, vol. 28, no. 2, pp. 167–182, 2005.
8. A. Abraham and J. Thomas, **Distributed intrusion detection systems: a computational intelligence approach,** *Applications of information systems to homeland security and defense*. *USA: Idea Group Inc. Publishers*, pp. 105–35, 2005.
9. Z.-H. Zhou. **Ensemble learning**. National Key Laboratory for Novel Software Technology, Nanjing University. Nanjing 210093, China. [Online].Available:http://cs.nju.edu.cn/zhouzh/zhouzh.files/publication/springerebr09.pdf.
10. G. Kumar and K. Kumar, **The use of artificial-intelligence-based ensembles for intrusion detection: a review,** *Applied Computational Intelligence and Soft Computing*, vol. 2012, p. 21, 2012.
11. S. Dzeroski and B. Zenko, **Is combining classifiers with stacking better than selecting the best one?** *Machine learning*, vol. 54, no. 3, pp. 255–273, 2004.
12. I. H. Witten and E. Frank, **Data Mining: Practical machine learning tools and techniques**. Morgan Kaufmann, 2005.
13. H. Kim, H. Kim, H. Moon, and H. Ahn, **A weight-adjusted voting algorithm for ensembles of classifiers,** *Journal of the Korean Statistical Society*, vol. 40, no. 4, pp. 437–449, 2011.
14. S. Chebrolu, A. Abraham, and J. P. Thomas, **Feature deduction and ensemble design of intrusion detection systems,** *Computers & Security*, vol. 24, no. 4, pp. 295–307, 2005.
15. A. Zainal, M. A. Maarof, S. M. Shamsuddin et al., **Ensemble classifiers for network intrusion detection system,** *Journal of Information Assurance and Security*, vol. 4, pp. 217–225, 2009.
16. Weka. **Weka.** *The University of Waikato*. [Online]. Available: http://www.cs.waikato.ac.nz/ ml/weka/
17. J. Tian. **Evaluation of classifers.** *Iowa State University*. [Online]. Available: http://www.cs.iastate.edu/~jtian/cs573/WWW/Lectures/lecture06-ClassifierEvaluation-2up.pdf.
18. KDD99. **KDDcup99**. *UCI Knowledge Discovery in Databases Archive*. [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.
19. S. J. Stolfo, W. Fan, W. Lee, A. Prodromidis, and P. K. Chan, **Costbased modeling for fraud and intrusion detection: Results from the jam project,** in *DARPA Information Survivability Conference and Exposition*, 2000. DISCEX'00. Proceedings, vol. 2. IEEE, 2000, pp.130–144.
20. KDD99. **Training attack types**. *UCI Knowledge Discovery in Databases Archive*. [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
21. G. K. Ahuja, K. K. Saluja, and M. Sachdeva, **An empirical comparative analysis of feature reduction methods for intrusion detection,** *International Journal of Information and Telecommunication Technology* (ISSN: 0976-5972), vol. 1, no. 1, 2010.
22. J. McHugh, **Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory**, *ACM transactions on Information and system Security*, vol. 3, no. 4, pp. 262–294, 2000.
23. M. Tavallaee, E. Bagheri, W. Lu, and A.-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.