INTRUSION DETECTION USING AN ENSEMBLE BASED MODEL

Shabnam Hassaniahari
Computer Science
Uniersity of Ottawa
Ottawa,Canada

Tina Yazdizadeh
Information Technology
Carleton University
Ottawa,Canada

Prudhvi Raj
Computer Science
University of Ottawa
Ottawa,Canada

Arunachalam Dhakshinamurthy
ComputerScience
Carleton University
Ottawa,Canada

***Abstract*— A massive amount of work has been conducted in the field of Intrusion Detection System (IDS) which is used to identify various attacks on the network. Various machine learning approaches have been carried out to prevent malware attacks or network intrusion by attackers. Single classifiers have several limitations which cause low performance in the classification of normal attacks and anomalies. In other words, they are not strong enough to create a good Intrusion Detection System (IDS). This is a reason researchers have the idea of building ensemble models to take advantage of different models in combination together. The main goal of using ensemble classifiers is to achieve higher accuracy and Lower False Alarm Rate (FAR) will be provided. Two ensemble learning techniques – Voting and Stacking based models are implemented with PSO optimizer to obtain weights.**

Keywords—Ensemble Classifier, Intrusion detection, NSL-KDD)

1. INTRODUCTION

In recent days, financial loss and crippled services have increased to many folds due to sophisticated cyber-attacks. As a result, traditional firewalls are not enough to protect networks against intrusions. We can leverage Intrusion Detection Systems (IDS) along with firewall to provide better security against intrusions. According to Rajadurai et al. [1] the IDS systems can be categorized in to four classes based on the detection approaches: 1. Signature based 2. Anomaly based 3. Host based and 4. Network based. The recent advancements in Machine Learning (ML) and Artificial Intelligence (AI) have helped to identify any type of network intrusions with ease. Among Machine Learning algorithms, single classifiers have been used extensively to approach intrusion detection problems. However, single classifiers might not fully result in the expected performance due to their inherent weaknesses for developing an IDS. Thus, more recent works have been taking advantages of combining different classifiers as base learners like ensembles in order to have lower FAR and better accuracy in terms of finding intrusions. To increase the robustness of IDS, in this research, five different types of classifiers have been used and they are categorized into strong and weak learners based on their performance. A novel ensemble approach is implemented to boost the performance of IDS by having the pair of weak and strong base learner, we use the Particle Swarm Optimization (PSO) for weighting the base learners and inject these weights to the ensemble classifiers.

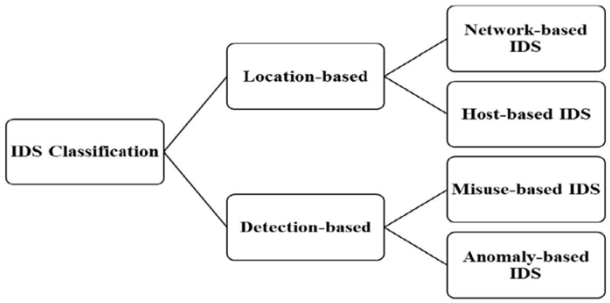

Figure 1. The IDS classification techniques

2. LITERATURE REVIEW

Among the several works which have been done in this field, Kumar et al. [2] studied in detail the methods which were used in this area. They described the ensemble learning methods as Bagging, Boosting and Stacking and categorized different learning algorithms in these categories. For example, they considered Random Forest and Wagging as Bagging (Bootstrap aggregating). Moreover, they have demonstrated a brief overview of ensemble integration methods such as Majority voting, Threshold Plurality Vote, Fuzzy theory-based, and Naïve Bayes. In the Majority voting, the output of each base classifier is interpreted as a vote for a specific class so that a class with the maximum number of votes will be the outcome of the ensemble classifier. They counted the Threshold Plurality Vote method as a generalized version of majority voting since it imposes a threshold on vote count for

choosing the class. Fuzzy theory-based method as the other integration method, employs fuzzy set theory for aggregating the outcomes of base classifiers and Naïve Bayes decision method considers conditional independence between the base classifiers.

The other survey by Sagi et al. [3]points out the major strengths and weaknesses of using an ensemble-based approach rather than using an individual learner. They talk about how an ensemble-based approach can overcome the overfitting avoidance by averaging out different hypotheses which will reduce the risk of choosing an incorrect hypothesis. They also focus on the edge that ensemble-based learner has over the single learner based on computational performance. In addition, they highlighted various unique advantages it has compared to the single learners and clearly mentioned how these ensemble learners can tackle the problem of class imbalance, concept drift and curse of dimensionality. The performance of a model can be improved using ensemble modelling which combines various machine learning techniques into a single predictive model. The majority of the ensemble methods fall under homogeneous learners. On the other hand, some methods use learners of different types called heterogeneous learners. Bagging, Boosting and Stacking are some of the commonly used ensemble-learning techniques. Overall, the paper showcases how these methods weigh several individual models and finally combine them to improve the predictive performance.

Abrar et al. [4] proposed a comprehensive approach to prevent unauthorized access to network resources and detect anomalies in the network. Several machine learning (ML) classifiers, namely Support Vector Machine (SVM), K-nearest neighbour (KNN), Logistic Regression (LR), Naïve Bayes (NB), Multi-layer Perceptron (MLP), Random Forest (RF), Extra-tree Classifier (ETC) and Decision Tree were used for the classification of data as normal or intrusive. Four feature subsets extracted from the NSL-KDD dataset are used as the train and target data. The performance of RF, ETC and DT was observed to be 99% for all the classes using different subsets of features. They did not use any optimization technique to improve the performance. An ensemble-based approach, which combines the output of various algorithms to predict the final results, could be the possible improvements in the paper.

On the other hand, Seth et al. [5] proposed a new approach for multiclass attack detection using the ensemble algorithms. Their approach ranked the detection ability of different base classifiers for identifying different types of attacks. The rankings are based on the final F1 score of each classifier for their predictions. The proposed approach is different from the voting approach because in the voting, the final predictions are based on the results of the majority vote among the selected models, and there is no control on whether the majority vote algorithm is efficient enough to detect the attack or not. CICIDS 2018 is their selected dataset, because it is the most recent dataset for the intrusion detection. This dataset is imbalanced and Seth et al. used a hybrid approach which used SMOTE for solving the imbalance issues. In addition to balancing the dataset, they detected the outliers in the dataset by using the Isolation Forest method which is an unsupervised machine learning algorithm based on Random Forest and Decision Tree. For the feature selection part, first, random forest is applied, then Principal Component Analysis (PCA) is used to eliminate the redundant features. Finally, they used 24 most significant features for the next steps. They used the seven most popular algorithms for the training phase which are: KNN, Decision Tree, Random Forest, Extra trees, XGBoost, Histogram-based Gradient Boosting, and Light GBM. Based on the performance of each classifier, a rank matrix was calculated and based on this matrix the results of the best classifier were used for the final prediction. The experiments showed that some of the classifiers have high recall but low precision, like RF, that is why Seth et al. used F1-score as the main metric. The results of the proposed method outperforms the single base classifiers' results. For example, the proposed method obtained 100% for detecting Bots and DDoS attacks.

Classification performance is improved with ensembles by the combined use of two effects .i.e., reduction of errors due to bias and variance. In his study, Govindarajan [6] proposed new ensemble classification methods with homogeneous ensemble classifiers using bagging and heterogeneous ensemble classifiers using arcing. Moreover, Radial Basis Function (RBF) and Support Vector Machine (SVM) are used as base classifiers in designing an ensemble classifier. The researcher's proposed approach is based on three main parts: pre-processing phase, classification phase, and combining phase, and performances are analysed in terms of accuracy. To know the performance of proposed homogeneous and heterogeneous classifiers, these are compared to the performance of other standard homogeneous classifiers such as Error-correcting output codes (ECOC), Bagging, and heterogeneous ensembles such as majority voting and stacking. Experimental results show proposed ensemble methods show better accuracy than individual classifiers. Furthermore, ECOC and Bagging are outperformed by the proposed RBF and SVM. Also, the proposed hybrid Radial Basis Function -Support Vector Machine performs significantly better than voting and stacking.

Another research work in this field by Bhati and Rai [7] proposed an ensemble-based approach for intrusion detection using extra tree classifiers. They combined the decision of different classifiers to increase the decisional power of the classifier. KDDcup99 and NSL-KDD datasets were used for the comparison as they are the benchmarks for intrusion detection. The proposed algorithm achieved 99.97% accuracy on the KDDcup99 dataset and 99.32 % on the NSL-KDD dataset. They did not implement any optimization approach to improve the performance of the model.

The aim of Pham et al. [8]work is to improve the performance of an IDS using the hybrid methods and feature selection.

They used the Bagging and Boosting which are the most popular ensemble techniques with a tree-based classifier as the base classifier. The performance of their model was evaluated on the NSL-KDD dataset and compared with the existing papers. Decision trees are their selected based classifier because it is sensitive enough to data resampling and feature resampling, and this is required for diversity in the ensemble model. Moreover, this classifier can apply on both nominal and numeric features with both continuous and discrete values. Finally, the tree-based algorithms can eliminate the irrelevant features as well as feature selection because of the use of the best variable in each split of the tree. Their selected tree-based algorithms are J48, Random Tree, REP Tree , Random Forest. As the feature selection techniques, they used Leave-One-Out for extracting 25 features and Gain Ration for extracting 35 features. The experiments showed that in 25 features set, although Bagging with REP Tree increased the accuracy, it didn't reduce the FAR. In this subset, the J48 had the lowest FAR. But in the 35 features subset, they achieved the goal of having more accuracy with less FAR by using Bagging with J48.

While Pham et al. [8]focused on ensemble and tree-based classifiers, Yousef Nezhad et al. [9]use deep learning in the feature selection phase in addition to using ensemble classification. Their main classifiers are SVM and KNN with different hyperparameters in eight versions. The SVM experimented with two different kernels which are RBF kernel valued 1 and 3 and Hermit Kernel with the degree of 8 and 10 for increasing the classification speed. They selected the RBF based on the experimental results which showed it has better performance. The KNN with 3, 5, 8, and 10 nearest neighbours is used for considering the diversity and better performance of the classifier. The output of SVM and KNN are converted to probable values using sigmoid function and heuristic algorithms respectively. Then, for integrating the data, they used Dempster-Shafer combining ensemble classification. This method can integrate the numerical, signals, and multidimensional data. Using the Dempster-Shafer is the final step in their procedure. They used an ensemble margin which is an important concept in ensemble learning, for selecting the samples from the KDD99 dataset. Then they used the new dataset in their proposed algorithm using the feature selection based on ensemble margin. Based on their results, using the Dempster-Shafer as an integrating module helped them in increasing the performance.

Another way of improving performance is to use the feature selection optimizers. In [10] Zhou et al. proposed a new ensemble approach on the basis of the modified adaptive boosting with the area under the curve (M-AdaBoost-A) algorithm so that network intrusions will be identified. In this work authors combined many M-AdaBoost-A-based classifiers to provide an ensemble classifier by employing various strategies such as simple majority voting (SMV) and Particle Swarm Optimization (PSO). In fact, the proposed approach, M-AdaBoost-A algorithm, takes into account the area under the curve into the boosting process to be able to address class imbalance issue in network intrusion detection. It utilized both the PSO algorithm and SMV to combine multiple M-AdaBoost-A-based classifiers into an ensemble with the aim of achieving improvements in network intrusion detection; thus, this system can be considered as a multiple-boosting- fused ensemble. In general, the M-AdaBoost-A-SMV ensemble is described as a set of multilevel nested models.

In addition to previous work, with irrelevant and redundant features, the problem of huge network traffic data size and the invisibility patterns have posed a great challenge in the domain of intrusion detection. Zainal et al. [11]proposed a way to address these challenges. In their paper, they employed two means; one is selecting the appropriate important features which represent the patterns of traffic and the second is forming an ensemble classifier model by engineering multiple classifiers with different learning paradigms. In their study, researchers formed a 2-tier selection process, by utilizing a hybrid approach where Rough Set Technique and Binary Particle Swarm (BPSO) were structured in a hierarchical manner. Here each class had one specific feature set since features were obtained based on class-specific characteristics. Moreover, Rough Set techniques were used to remove the redundant features and rank the top 15 features for each class of traffic since BPSO uses the heuristic technique. These significant features are termed as reducts. In their study, researchers considered the decision function by taking the degree of diversity among the classifiers into account. To classify the network connection, ensemble machine learning techniques with different learning paradigms .i.e. Linear Genetic Programming(LGP), Adaptive Neural Fuzzy Inference System(ANFIS) and Random Forest(RF) were used and decision function was determined based on the individual performances on overall accuracy and true positive rates. Furthermore, the experimental results show an improvement in detection accuracy for all classes of network traffic.

Nenekazi et al. [12]proposed a performance bound of a network intrusion detection that uses an ensemble of classifiers. Till now, all the research focussed on implementing the network intrusion detection that leverages ensemble-based classifiers without knowing the performance of their intrusion detection systems. With the proposed model, researchers will come to know the performance of their ensemble-based NIDS even without implementing it. For this study, researchers used the NSL-KDD dataset that was filtered for normal and Neptune connections. In this study, researchers have used classification accuracy as a performance measure for this ensemble and average information gain associated with features to define the bound. In addition, the Adaboost algorithm was used to obtain performance bound in order to boost the ensemble performance to converge to its optimality. Moreover, the performance of an Adaboost based network intrusion detection system that uses a decision stump as a weak learner to classify Neptune and normal connections can be estimated by the bound. From the experimental results,

researchers concluded that the accuracy of the ensemble will be at most 0.9 if the information gain value amongst features used in the ensemble lies between 0.04561 and 0.25615.

3. DATASET

NSL – KDD is a dataset suggested to solve some of the inherent problems of the KDD'99 data set and contains the records of the internet traffic seen by a simple intrusion detection work. This dataset comprised of two general sub datasets: KDD_Train+ and KDD_Test+. In this project we used KDD_Train+ because it prevents the overfitting of the model. The dataset consists of 42 features and out of which 41 corresponds to the traffic input and the last one corresponds to the class label. Again, these features can be categorized into four categories such as Intrinsic, Content, Host-based, and Time-based.

Intrinsic Features: These features contain the packet information and can be derived from the header of the packet. This category contains features 1-9.

Content Features: These features contain original packets information which can be used by the system to access the payload. This category contains features 10-22.

Time-based features: These features contain the information about how many connections it attempted to make to the same host over 2 two-second windows. This category contains features 23-31.

Host-based features: These features contain information about how many requests are made to the same host over x-number of connections. This category contains features 32-41.

Also, the feature types in this data set can be broken down into 4 types which are categorical, binary, discrete, and continuous. The distribution of the data types is as follow: 4 Categorical, 6 Binary, 23 Discrete, and 10 Continuous. The categorical features are Protocol Type, Service, Flag. Among these columns, Flag values is not easy to understand. This feature describes the status of the connection showing whether there is a flag raised or not. Since machine learning algorithms cannot operate on label data directly, we will convert the categorical values into numerical values.

The reasons behind using this dataset in this project are first during the literature review, most of the researchers mentioned that they used NSL-KDD as it contains complete information about intrusions inside the internet traffic. Moreover, this dataset does not have redundant records in the train data, so the classifier will not provide any biased results during prediction.

4. METHODOLOGY

In the current project, after data preprocessing step and extracting the features of the data, 5 different machine learning algorithms are chosen and trained on the dataset. After training, they classified in to weak and strong learners based on their performance. Then by using Particle Swarm Optimization (PSO), average weights for the base learners calculated. The next step is to take advantage of two ensemble models namely, Stacking and Majority Voting. In this section, the base learners and the ensemble models will be described briefly. Then procedure of using PSO will be explained in detail. An overview of our methodology is shown in Figure 2.

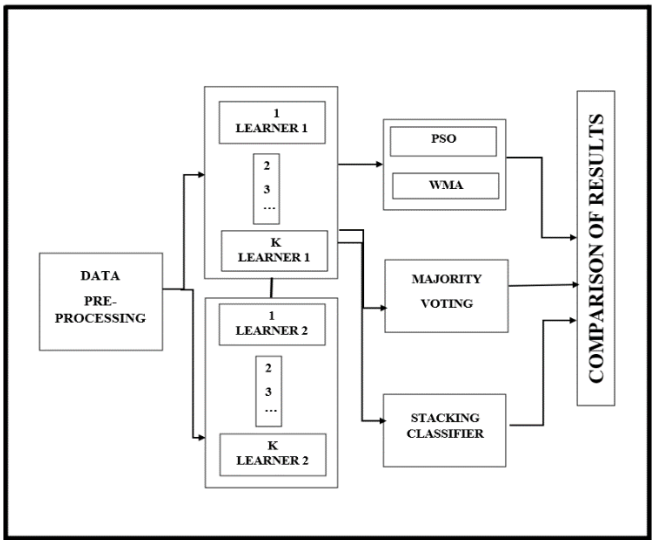

Figure 2. Methodology

4.1 Data Pre-Processing

As a very first step of each machine learning process, data preprocessing should be done in an effective way. Normally in this step, the categorical data is converted into the numerical type of data in order to have better results in the predictions. For this aim, we use one hot encoding as a method. We applied this method on 3 categorical columns which exist in the dataset namely, Protocol Type, Service and Flag. Besides, we encoded the 'Label' column to 0 and 1 by putting 1 instead of the anomaly and 0 instead of the normal. In addition to encoding the categorical data, since there are different values in the dataset, we applied data normalization for increasing the cohesion of the dataset. Basically, the data normalization means the reorganization of data to appear similar in whole dataset. This sub step is important because data normalization gets rid of the anomalies across the records and fields that can make the process of prediction complicated. Moreover, this process can make the dataset keep less space. For the data normalization, we used **"StandardScalar"** package from "**sklearn.preprocessing"** library. The final sub step is splitting the dataset into 70% train and 30% test because we want to see how the model works on the seen samples.

4.2 Feature Selection

The second step of the methodology is feature selection. The feature selection refers to techniques that make a subset of most relevant and important features of a dataset. Having fewer features allows the machine learning algorithm occupies less space and runs without time complexity. We used Random Forest as a method for extracting the important features. This method is one of the embedded methods. The embedded methods are implemented by algorithms that have their own built-in feature selection methods. We used one of the embedded methods, Random Forest, because these methods are highly accurate and interpretable. In the random forest, each tree of the random forest is a decision tree and can calculate the importance of feature based on the feature's ability in increasing the pureness of the leaves. With this method we collected the importance of features. Then we used Recursive Feature Elimination (RFE) as a wrapper-based feature selection. It means that a machine learning algorithm is used as a core and the RFE acts as a wrapper. The idea of the RFE is to select features by recursively considering the smaller and smaller sets of features. This method has two main steps which are: first, an estimator will be trained on a set of features and calculate the importance of each feature, and second step is to prune the least important features from the current set of features. This method is easy to configure and use. In this project the core machine learning algorithm is Random Forest as mentioned before. We first plot the feature importance which obtained by the Random Forest as shown in Figure 3. And then the total number of features which selected by RFE is 65.

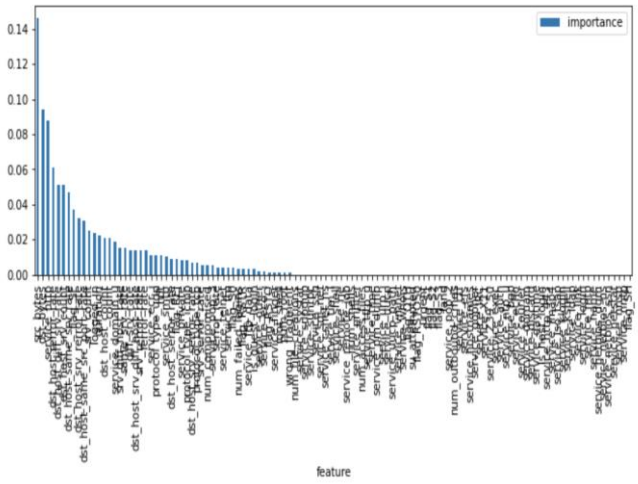

Figure 3. The Features Importance

4.3. Base Learners

The third step in our proposed method is to select and train 5 base learners. We used Support Vector Machine, Naïve Bayes, K-nearest Neighbor, Decision Tree, and Logistic Regression as our base learners. The base learners are selected based on the literature study and discovering the strengths and weaknesses of each classifier. In the following a brief description of each of the base learners is explained. The results of each base learner are explained in the Section 7 in detail.

4.3.1 Support Vector Machine

Support Vector Machine (SVM) each data is plotted as a point in n-dimensional space where n is the number of features. The classification is done by finding a hyper plane that distinguish the two classes in a good way. This classifier can address space complexity and it further creates non-linearity and least impacted by outliers. The performance of SVM is related to the hyperparameter tuning. In this project we use SVC with enabling the probability estimates.

4.3.2 Naïve Bayes

Naïve Bayes Classifier is one of the simple and most operative Classification algorithms. It is easy and fast to predict class of test data set. This classifier is a probabilistic one, which means it predicts based on the probability of an object. Naïve Bayes classifier performs better compared to other models like logistic regression when the problem has independence. Naïve Bayes classifiers are based on Bayes theorem and has the potential to achieve high accuracy and data interpretability. We used the GaussianNB among the different kinds of the Naïve Bayes because it can be used in the online learning in future.

4.3.3 K-nearest Neighbor

K-Nearest Neighbor is a supervised classifier. Initially, the value of K is selected, and then Euclidean distance is computed among the various data points based on which the data is separated into K-number of clusters. KNN is easy to interpret outputs and has low computation time. In this work, we use a KNeighborclassifier from sckit learn package with default parameters. The number of neighbors we used 1,3,5,7,9, and 11.

4.3.4 Decision Tree

The goal of using Decision Tree is to create a training model that can use to predict the class or value of the label by learning simple decision rules inferred from previous data (training data). Decision tree used tree like models to make decisions and entirely based on control if statements. This classifier can observe information and identify critical qualities in the system that demonstrate the malicious activities like the intrusions. In this project we used the sckit learn decision tree as one of the best algorithms for intrusion detection task. The max depths used in the ensemble part are 2, 4, 6, 8, 10, and 12. The result of this base learner is reported in Results section along with other learners.

4.3.5 Logistic Regression

Logistic Regression is used to model the probability of certain class or event. It is used for the independence and interpretability of data. This statistical model is easy to implement and very efficient to train. By the way, if the number of observed features is less than the number of features this classifier is not appropriate for classification. We used the Logistic regression with the maximum iteration of 100 for the solvers to converge. As the maximum iterations for the ensemble part we used 100,10,1,0.1,0.01, and 0.001.

4.4 Ensemble Models

An ensemble classifier is a method which uses or combines several classifiers to develop robustness of a system as well as improving the performance from any of the basic classifiers. Based on Schapire [13]and Dong et al. [14], Ensemble methods have the improvement on what they can be made to adapt to any changes in the monitored data stream more accurately than single model techniques. As an important parameter for the success of an ensemble approach, diversity in the individual classifiers with respect to misclassified instances can be mentioned. Dietterich [15] reported that there are three main reasons why an ensemble classifier is usually significantly better than a single classifier. First of all, the training data does not always deliver enough information for selecting a single accurate hypothesis. Moreover, the learning processes of the weak classifier could be imperfect. In addition, the hypothesis space being searched might not contain the true target function while an ensemble classifier can provide a good estimate. Ensemble learning has three types which are bagging, boosting, and stacking. Bagging and boosting are the alternatives of the voting methods. In this project, after training the base learners, we used two different Ensemble Learners, Majority Voting and Stacking. In the following, a description and the way of using these classifiers are explained in detail.

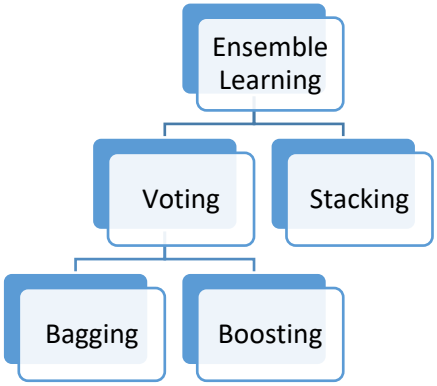

Figure 4. Ensemble Classification Techniques

4.4.1 Particle Swarm Optimization

Particle swarm optimization is a population-based iterative optimization algorithm, formulated by Kennedy and Eberhart. PSO is derivative-free, zero-order method. That means it does not need gradients, so it can be applied to a variety of problems, including those with discontinuous or non-convex and multimodal problems. The algorithm starts out with a set of agents, called particles, in random positions in the problem space. Each is also assigned random velocity at the outset. A fitness function is defined on a particle's location. The optimization problem to be solved is to find the best position, i.e. the one that minimizes the fitness function. Through each iteration, the algorithm evaluates each particle's fitness, updates its velocity, and computes its new position. A particle's new velocity depends on its current velocity, its distance from its own best position so far and its distance from the populations best position yet. Compared to genetic algorithms (GA), PSO has no evolution operators such as crossover and mutation which makes it easy to implement with great success to several problems wherever GA can be applied.

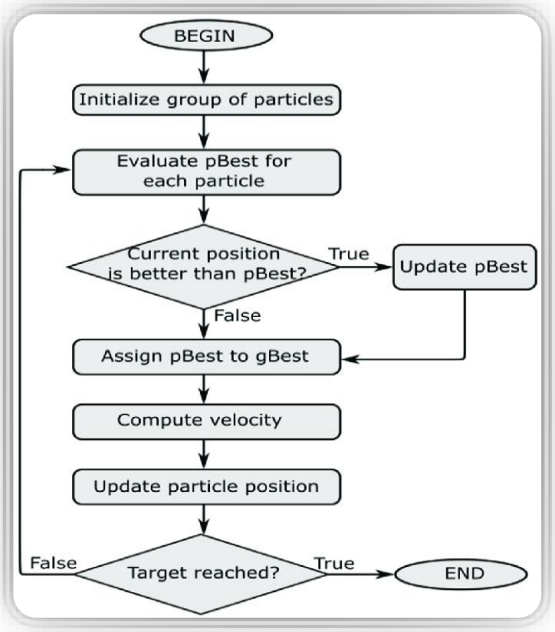

Figure 5. The Flow Chart for PSO

The weights obtained from PSO are chosen as the weights for the learners w [0:12].

4.4.2 Majority Voting

A very big drawback of Voting Classifier is that it assumes that all models in the ensemble are equally effective. This may not be the case as some models may be better than others especially if different machine learning algorithms are used to train each model ensemble member. An alternative to voting

is to assume that ensemble members are not all equally capable and instead some models are better than others and should be given more votes when making a prediction. This provides the motivation for the weighted average ensemble method.
Weighted average ensembles allow the contribution of each ensemble member to a prediction to be weighted proportionally to the trust or performance of the member on a holdout dataset. Model averaging is an approach to ensemble learning where each ensemble member contributes an equal amount to the final prediction. In the case of predicting a class label, the prediction is calculated as the mode of the member predictions. In the case of predicting a class probability, the prediction can be calculated as the argmax of the summed probabilities for each class label.

To ensure greater diversity in the classifiers and to potentially maximize the use of them, each classifier is trained with 6 different parameter and then combined with another classifier with 6 different parameters to utilize the full potential of these classifiers. For example, in case of KNN-Logistic Regression ensemble model, 6 KNN models with value of K tuned for each model are combined with 6 logistic regression model with value of C tuned for each model.

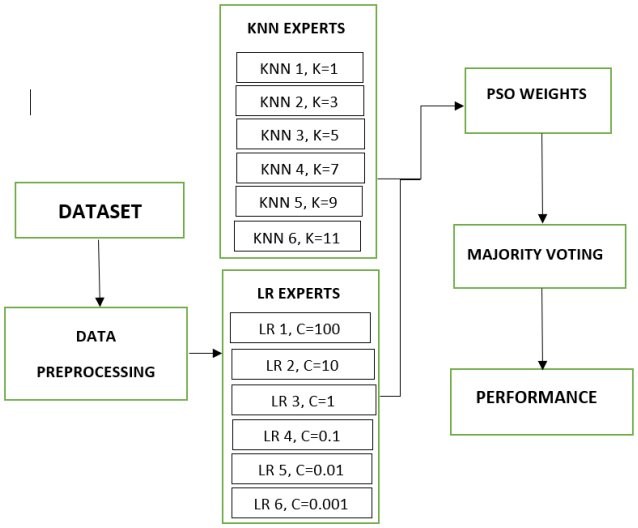

Figure 6. The Flow Chart for Ensemble Models

Similarly, all models are trained and tested with this novel ensemble method.

4.4.3 Stacking

Stacking, stacked generalization, is a different technique of combining multiple classifiers. Unlike bagging and boosting, stacking is usually used to combine various classifiers, e.g. decision tree, neural network, naïve bayes, logistic regression, and etc. This method is an effective approach as it is a general framework, which combines many ensemble methods. The learning process in this method consists of two levels, base learning and meta-learning. In base learning, the initial (base) learners are trained with training data set in order to create a new dataset for the next step which is the meta-learning. The meta-learner is trained with new training data set. The trained meta-learner is used to classify the test set. A crucial part in stacking is the selection of a best base learner. In this project we choose the base algorithms which are frequently used in the literature review for intrusion detection and then classified them into weak and strong based on their performance. The final estimator in all stacking models is Logistic Regression.

5. EVALUATION

Model evaluation is the subsidiary part of the model development process. In this phase, whether the model performs better or not is investigated. Following the different evaluation metrics that used in this project are described. Precision, Recall, Accuracy, F1 score, Specificity, and Sensitivity are the metrics calculated in this project, but based on F1 score final decisions are made. All the metrics are based on a confusion matrix which made of the ratio of true or false predictions to observed samples. A confusion matrix helps us gain an insight into how correct our predictions were and how they hold up against the actual values.

5.1 Precision

Precision quantifies the number of positive class predictions that belong to the positive class. In other words, it is the ratio of the True Positives to all the observed Positives predictions. Therefore, higher precision means the model predicted attacks more accurately and if it had doubts about an instance, only with a low probability did it consider as an attack (low False Positive leads to high Precision)

$$\text{Precision}=\frac{TP}{TP+FP}$$

5.2 Recall

Recall quantifies the number of positive class predictions made out of all positive examples in the dataset. Higher recall means the model tried well in assigning the label attack to those instances that were actually an attack; therefore, the system with a high value of recall metric, will be more secure since it detects attacks with higher chances.

$$\text{Recall}=\frac{TP}{TP+FN}$$

5.3 F1 Score

F1-score considers both precision and recall metrics; so it is a good indicator of how the model performs. In fact, The F1

score can be interpreted as a harmonic mean of the precision and recall metrics, where an F1 score reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall to the F1 score are equal.

$$\text{F1-Score}=\frac{2*Precision*Recall}{Precision+Recall}$$

5.4 Accuracy

Accuracy is the fraction of predictions that the model got right since it is calculated as the ratio of all true predictions to all the possible predictions.

$$\text{Accuracy}=\frac{Number\ of\ Correct\ Predictions}{Total\ Number\ of\ Predictions}=\frac{TP+TN}{TP+TN+FP+FN}$$

5.5 Specificity

Specificity is the metric that evaluates a model's ability to predict true negatives of each available category. Specificity is defined as the proportion of actual negatives, which got predicted as the negative (or true negative). This implies that there will be another proportion of actual negative, which got predicted as positive and could be termed as false positives. This proportion could also be called a false positive rate. It is worth. Mentioning that while sensitivity measure is used to determine the proportion of actual positive cases, which got predicted correctly, Specificity measure is used to determine the proportion of actual negative cases, which got predicted correctly.

$$\text{Specificity}=\frac{TN}{TN+FP}$$

5.6 Receiver Operator Characteristic (ROC)

The Receiver Operator Characteristic (ROC) curve is an evaluation metric for binary classification problems. It is a probability curve that plots the TPR against FPR at various threshold values and essentially separates the 'signal' from the 'noise'. Also, the Area Under the Curve (AUC) is the measure of the ability of a classifier to distinguish between classes and is used as a summary of the ROC curve. The higher the AUC, the better the performance of the model at distinguishing between the positive and negative classes.

5.7 Geometric Mean

The geometric mean is calculated as the N-th root of the product of all values, where N is the number of values. The geometric mean accepts positive inputs and not the zero. Geometric mean in machine learning is in the calculation of the so-called G-Mean (geometric mean) metric that is a model evaluation metric that is calculated as the geometric mean of the recall and specificity metrics. This metric can be calculated using the Scipy package and gmean function.

6. RESULTS

As discussed before, various metrics have been calculated to assess the performance of our models. Generally, if the data set is balanced, we can use accuracy metric to assess the model's performance. If the dataset is not balanced, then choosing accuracy as a metric is not ideal because it does not distinguish between the numbers of correctly classified examples of different classes. So, in future even though different dataset is used, to compare our model's performance we are choosing F1 score metric as base metric to evaluate the performance of the models.

Model / Measure	Decision Tree	SVM	LR	Naive Bayes	KNN
Accuracy	0.98255	0.96792	0.95801	0.90686	0.973536369
Precision	0.9835	0.95983	0.94544	0.96384	0.974895
Recall	0.98557	0.98426	0.98216	0.86723	0.978221
F1 score	0.98454	0.97189	0.96345	0.91298	0.976555
Specificity	0.9835	0.95983	0.94544	0.96384	0.974895397
- Predictive val	0.98557	0.98426	0.98216	0.86723	0.978220939
+ Predictive val	0.97867	0.94683	0.92685	0.95801	0.967490687
Geometric Mean	0.98241	0.96936	0.96048	0.90421	0.973330797
AUCROC Value	0.98262	0.99536	0.98207	0.94987	0.993251

Table 1. The results of Base learners

Above Table represents the results of our baseline models. From the results we can infer that Decision Tree is the best performing model with the F1-score of 0.98. It closely followed by SVM, K-Nearest Neighbor, Logistic Regression and Naïve Bayes. Among all these models, Naïve Bayes is the worst performing model with the F1-score of 0.91. Below figure represents the ROC plots of our baseline models.

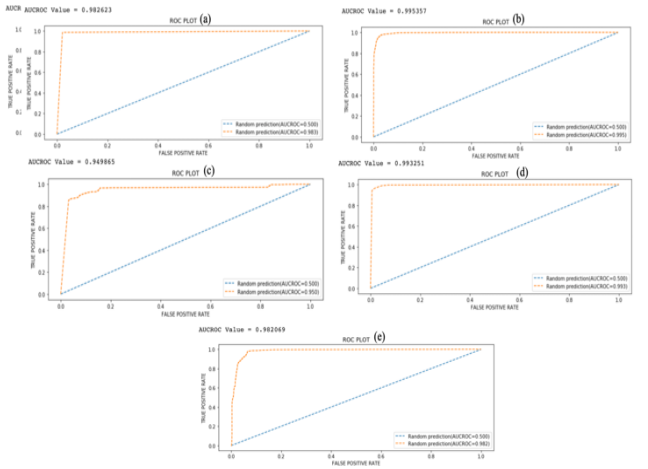

Figure 6. ROC plots for Base Learners; (a) Decision Tree (b) SVM (c) Naïve Bayes (d) KNN (e) Logistic regression

After assessing the performance of base classifiers and categorizing them into strong and weak classifiers, we proceeded with building ensemble models by voting and stacking.
Table 2. represents the performance metrics of various ensembles models that are built by using voting method.

Measure \ Model	(LR, DT)	(LR, KNN)	(DT, KNN)	(SVM, KNN)	(SVM, DT)	(SVM, LR)
Accuracy	**0.965**	**0.944**	**0.960**	**0.946**	**0.961**	**0.962**
Precision	**0.961**	**0.941**	**0.957**	**0.957**	**0.954**	**0.956**
Recall	**0.978**	**0.961**	**0.972**	**0.981**	**0.979**	**0.979**
F1 score	**0.969**	**0.951**	**0.954**	**0.969**	**0.966**	**0.967**
Specificity	**0.962**	**0.941**	**0.957**	**0.957**	**0.954**	**0.956**
- Predictive val	**0.978**	**0.961**	**0.972**	**0.981**	**0.979**	**0.979**
+ Predictive val	**0.949**	**0.922**	**0.944**	**0.942**	**0.932**	**0.941**
Geometric Mean	**0.966**	**0.945**	**0.960**	**0.966**	**0.963**	**0.963**
AUCROC Value	**0.991**	**0.966**	**0.983**	**0.99**	**0.981**	**0.983**

Table 2. The results of Ensemble Learners (Majority Voting)

From the above Table, we can infer that (SVM, KNN) is the best performing ensemble model followed by (Logistic Regression, Decision Tree), (SVM, Logistic Regression), (SVM, Decision Tree), (Logistic Regression, KNN) and (Decision Tree, KNN). Also, F1-score metric is used to assess the performance of the models. Furthermore, the performance of ensemble models (Voting) is lesser than the baseline models.

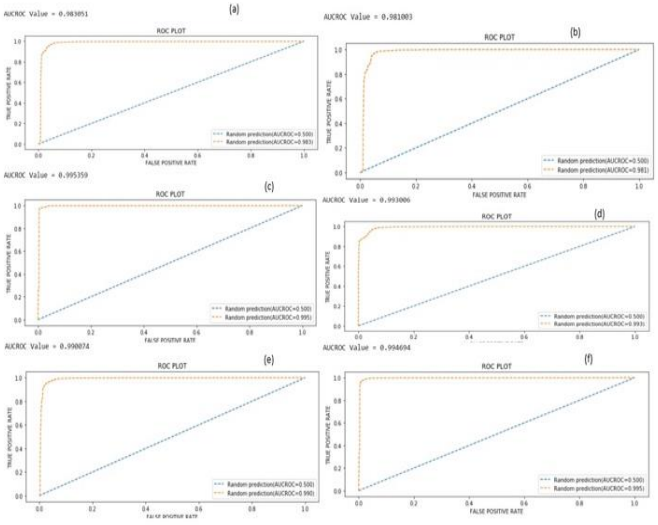

Figure 7. ROC plots for Majority Voting; (a) SVM&LR (b) SVM&DT (c) DT&KNN (d) KNN&LR (e) SVM&KNN (f) DT&LR

Furthermore, we built ensembles using stacking and assessed the performance of the models. To assess the performance, we considered F1-score as base metric.

Measure \ Model	(LR, DT)	(LR, KNN)	(DT, KNN)	(SVM, KNN)	(SVM, DT)	(SVM, LR)
Accuracy	**0.973**	**0.984**	**0.972**	**0.983**	**0.973**	**0.984**
Precision	**0.977**	**0.983**	**0.971**	**0.979**	**0.975**	**0.985**
Recall	**0.976**	**0.989**	**0.979**	**0.992**	**0.977**	**0.986**
F1 score	**0.976**	**0.986**	**0.975**	**0.985**	**0.976**	**0.986**
Specificity	**0.977**	**0.983**	**0.971**	**0.978**	**0.975**	**0.985**
- Predictive val	**0.976**	**0.989**	**0.070**	**0.992**	**0.977**	**0.986**
+ Predictive val	**0.970**	**0.978**	**0.962**	**0.972**	**0.967**	**0.981**
Geometric Mean	**0.973**	**0.984**	**0.972**	**0.984**	**0.972**	**0.984**
AUCROC Value	**0.997**	**0.998**	**0.995**	**0.997**	**0.997**	**0.999**

Table 3. The results of Stacking Learners

From the above results, we can observe that (SVM, Decision Tree) and (Decision Tree, K-Nearest Neighbour) has same F1-scores followed by (Decision Tree, Logistic Regression), (SVM, K-Nearest Neighbour), (Logistic Regression, K-Nearest Neighbour) and (SVM, Logistic Regression). Even though (SVM, DT) and (DT, KNN) have same F1-score, we will choose (DT, KNN) is the best performing model because of the complexities involved with (SVM, DT) model. Furthermore, ensembles by stacking performed better than baseline models and ensembles by voting. One possible reason for this would be stacking allows to use the strength of each individual estimator by using their output as input of final estimator whereas voting classifier takes the most common output to be the output of final estimator.

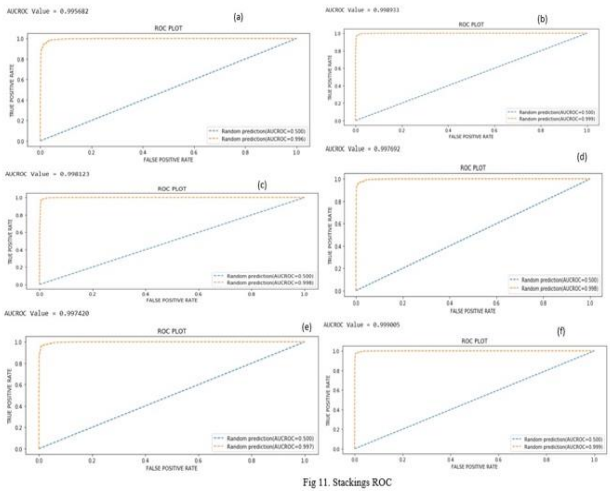

Figure 8. ROC plots for Stacking Ensembles; (a) SVM&LR (b) SVM&DT (c) LR&DT(d) SVM&KNN (e) LR&KNN (f) DT&KNN

7. COMPETITOR MODELS

We have critically compared and analyzed the results from literature survey with our proposed methodology. The comparison table is as follows,

Research Papers	Dataset used	Model used	F1-Score (%)	Accuracy (%)	Precision (%)
[10]	NSL-KDD	M-AdaBoost using PSO	91.64	99.89	88.34
[7]	NSL-KDD	Ensemble Trees Classifier	97.8%	-	99%
[16]	NSL-KDD	Majority Voting – (SVM, ETC, KNN, MLP, RF)	87%	92%	-
Proposed model	**NSL-KDD**	**Majority Voting with PSO**	**96.9%**	**96.5%**	**96.1%**
Proposed Model	**NSL-KDD**	**Stacking method**	**98.6%**	**98.4%**	**98.3%**

Table 4. Comparison of Competitor Models

Compared to all the baselines, the proposed method using majority voting with PSO and stacking method performed better in terms of F1-score, Accuracy and Precision. Primarily, our models were trained and tested using the best hyper-parameters for each ensemble learning. So, a greater diversity is obtained and the use of each classifier in the ensemble model is maximized by using the approach of combining 6 LRs and 6 KNNs in our approach.

The main drawback of using m-AdaBoost method [10], which is similar to our approach, did not perform better because of the problems related to overfitting the training data and failed to address the outliers.

8. LIMITATIONS AND ADVANTAGES

Generally, IDS has its own limitations. For example, most IDS do not process encrypted packets which results in the intrusion of network. In case of real-world intrusions, the number of false alarms is more compared to the number of real attacks which may lead to misclassification of real attacks.

Although we used a clean dataset, as a limitation in the intrusion detection system, we can mention the noises in a dataset. The bad packet generated from different sources like software bugs can severely limits the effectiveness of systems. The hyper parameter tuning of SVM model was time-consuming and more powerful hardware setup was required especially while training large datasets.

As a learning lesson from the project, not always a combination of strong base learners would perform better than the others (combination of weak learners). Also, it is not necessary for ensembles to perform better than the base learners all the time. It will be affected by the way, how this vase learners are combined to form the ensembles. For instance, in our case stacking performed better than voting classifiers because stacking allows us to use the strength of each individual estimator by using their output as the input of final estimator. Contrastingly, voting classifiers takes the most common output to the output of the final estimator which will work better only if we have multiple good models.

9. CONCLUSION AND FUTURE WORK

In this project we used 5 different base learners to have pairs of weak and string classifiers and then by using these pairs we performed classification with two ensemble learning models which are Majority Voting and Stacking. Moreover, we used Particle Swarm Optimization (PSO) as a mean for optimizing the weights of the ensemble models. Based on the results, stacking based ensembles outperformed voting based ensembles and the baseline models. Furthermore, out of all models, stacking based Decision Tree and K-nearest Neighbor ensemble model performed better with an F1 score of 98.6%. For the future works, using online ensemble learning can be used to cover more data. Despite using the static data, we want to adapt the model with data streams while considering the concept drift. Moreover, there are always open problems in the feature selection step to make the predictions better. Furthermore, using PSO with LUS optimization will obtain better weights.

References

[1] H. Rajadurai and U. D. Gandhi, "A stacked ensemble learning model for intrusion detection in wireless network," *Neural Comput. Appl.*, May 2020, doi: 10.1007/s00521-020-04986-5.

[2] G. Kumar, K. Thakur, and M. R. Ayyagari, "MLEsIDSs: machine learning-based ensembles for intrusion detection systems—a review," *J. Supercomput.*, vol. 76, no. 11, pp. 8938–8971, Nov. 2020, doi: 10.1007/s11227-020-03196-z.

[3] O. Sagi and L. Rokach, "Ensemble learning: A survey," *WIREs Data Min. Knowl. Discov.*, vol. 8, no. 4, Jul. 2018, doi: 10.1002/widm.1249.

[4] I. Abrar, Z. Ayub, F. Masoodi, and A. M. Bamhdi, "A Machine Learning Approach for Intrusion Detection System on NSL-KDD Dataset," in *2020 International Conference on Smart Electronics and Communication (ICOSEC)*, Trichy, India, Sep. 2020, pp. 919–924. doi: 10.1109/ICOSEC49089.2020.9215232.

[5] S. Seth, K. K. Chahal, and G. Singh, "A Novel Ensemble Framework for an Intelligent Intrusion Detection System," *IEEE Access*, vol. 9, pp. 138451–138467, 2021, doi: 10.1109/ACCESS.2021.3116219.

[6] M. Govindarajan, "Evaluation of Ensemble Classifiers for Intrusion Detection," vol. 10, no. 6, p. 9, 2016.

[7] B. S. Bhati and C. S. Rai, "Ensemble Based Approach for Intrusion Detection Using Extra Tree Classifier," in *Intelligent Computing in Engineering*, vol. 1125, V. K. Solanki, M. K. Hoang, Z. (Joan) Lu, and P. K. Pattnaik, Eds. Singapore: Springer Singapore, 2020, pp. 213–220. doi: 10.1007/978-981-15-2780-7_25.

[8] N. T. Pham, E. Foo, S. Suriadi, H. Jeffrey, and H. F. M. Lahza, "Improving performance of intrusion detection system using ensemble methods and feature selection," in *Proceedings of the Australasian Computer Science Week Multiconference*, Brisband Queensland Australia, Jan. 2018, pp. 1–6. doi: 10.1145/3167918.3167951.

[9] M. Yousefnezhad, J. Hamidzadeh, and M. Aliannejadi, "Ensemble classification for intrusion detection via feature extraction based on deep Learning," *Soft Comput.*, vol. 25, no. 20, pp. 12667–12683, Oct. 2021, doi: 10.1007/s00500-021-06067-8.

[10] Y. Zhou, T. A. Mazzuchi, and S. Sarkani, "M-AdaBoost-A based ensemble system for network intrusion detection," *Expert Syst. Appl.*, vol. 162, p. 113864, Dec. 2020, doi: 10.1016/j.eswa.2020.113864.

[11] A. Zainal, M. A. Maarof, and S. M. Shamsuddin, "Ensemble Classifiers for Network Intrusion Detection System," p. 10.

[12] N. N. P. Mkuzangwe, F. Nelwamondo, N. N. P. Mkuzangwe, and F. Nelwamondo, "Ensemble of classifiers based network intrusion detection system performance bound," in *2017 4th International Conference on Systems and Informatics (ICSAI)*, Hangzhou, Nov. 2017, pp. 970–974. doi: 10.1109/ICSAI.2017.8248426.

[13] R. E. Schapire, "The Boosting Approach to Machine Learning: An Overview," in *Nonlinear Estimation and Classification*, vol. 171, D. D. Denison, M. H. Hansen, C. C. Holmes, B. Mallick, and B. Yu, Eds. New York, NY: Springer New York, 2003, pp. 149–171. doi: 10.1007/978-0-387-21579-2_9.

[14] Yan-Shi Dong and Ke-Song Han, "A comparison of several ensemble methods for text categorization," in *IEEE International Conference onServices Computing, 2004. (SCC 2004). Proceedings. 2004*, Shanghai, China, 2004, pp. 419–422. doi: 10.1109/SCC.2004.1358033.

[15] T. G. Dietterich, "Machine-Learning Research," p. 40.

[16] A. M. Bamhdi, I. Abrar, and F. Masoodi, "An ensemble based approach for effective intrusion detection using majority voting," *TELKOMNIKA Telecommun. Comput. Electron. Control*, vol. 19, no. 2, p. 664, Apr. 2021, doi: 10.12928/telkomnika.v19i2.18325.