NIDS-HCTWSVM: A Network Intrusion Detection System Based on Hierarchical Clustering and Twin Support Vector Machine

Harini Induri[1], Dr. M. Dhanalakshmi[2]
[1]*University College of Engineering, Science and Technology, Hyderabad, JNTUH*
[2]*Professor of IT, Jawaharlal Nehru Technological University, Hyderabad*

***Abstract*—Technology is upgrading day by day. With the improving technology, the security issues are also becoming challenging. As the number of people using network are increasing, the data will also get increases. It's necessary to know if any person is entering our network which can give the attacker a chance to steal and misuse data. To handle this kind of issues, having an intrusion detection system is really necessary which can detect and give output of network traffic if it's normal or not. This study first proposes a solution for designing network intrusion detection system by combining hierarchical clustering, Decision tree, Random Forest, and, twin support vector machine, that can be effectively used to detect various classes of network intrusion. Apart from that, different ML models like SVM, Random Forest, gradient boosting and so on are also trained to check the efficiency of the proposed system. The designed system will detect if the network traffic is Normal or not. If it's an attack it also specifies the type of attack. In Combined model, hierarchical clustering algorithm will be applied for making network traffic uniform for training. After implementing the agglomerative clustering, a decision tree is constructed to reach the goal of reducing the error accumulation while constructing the decision tree. Finally, based on the above results, a network intrusion detection model is realized by incorporating twin support vector machines in the decision tree built, which detects the category of network intrusion.**

***Index Terms*—Decision Tree, Gaussian Naïve Bayes, Gradient boosting, Hierarchical clustering, IDS, Random Forest, SVM, Twin SVM.**

I. INTRODUCTION

Data innovation (IT) is a broader field that incorporates cybersecurity, whereas cybersecurity is a specialized component of IT security. IT experts utilize computers and other innovation to store, recover, and oversee data. IT moreover incorporates IT security, which ensures touchy data from dangers, and cybersecurity, which centers on keeping computer frameworks, systems, and information secure from cyberattacks. With the expanding enhancement of computer systems and communication systems, the internet security is facing complex and different dangers of security such as arrange interruptions and many attacks. IDS is an watching framework that identifies suspicious exercises and produces alarms when they are identified and executed in conjunction with security concerns and strategies such as confirmation, security framework and encryption approaches to fortify security against cyber-attacks. Due to the dangerous development of organize activity, planning effective and strong organize interruption discovery frameworks in a huge information environment has gotten to be significant but troublesome. Subsequently, it is vital to ponder quick and precise brilliantly organize interruption discovery strategies to guard against different organize interruptions in complex arrange environments. So, in this paper, we proposed an IDS which is a combination of Hierarchical clustering, decision tree/ Random Forest and Twin SVM which can accurately detect for attacks in network traffic.

In the proposed system, the designed IDS, not only classifies the data, but also it detects the category of attack which would be a multi class classification problem.

Hence we tried to implement, IDS using only TWSVM but it gave low accuracy, so tried an experimental approach of combining multiple models. Designed IDS for combined Decision tree, Hierarchical clustering and Twin svm and also for combination of Random forest, Hierarchical clustering and Twin svm.

Also tested the designed IDS with traditional ML models like SVM, Decision tree, Gaussian Naïve bayes, Random Forest, and Gradient boosting.

II. OBJECTIVES OF THE STUDY

A. Primary Objective

An Intrusion detection system need to be designed which detects if network traffic is normal or abnormal.

B. Secondary Objectives

1. Preprocessing the data set
2. Train the combined models(Decision tree+TWSVM, Rand+TWSVM)
3. Train the Traditional models(SVM, Rand Forest, Gradient Boosting, Decision tree, Naïve Bayes)
4. Get the results
5. Performance evaluation based on accuracy.

III. LITERATURE SURVEY

Artificial intelligence as been developing day by day. With the development in technology, various ML algorithms are also proposed to construct Intrusion detection systems.

Applying advanced techniques of ML to design an IDS is one of the important paths where we have scope for research.

According to the ML algorithms used in the design, the IDS are divided into 2 types. They are:

1. Traditional ML based IDS
2. Deep learning bases IDS

The traditional method design will be simple and the computation time will be low which can be used in the real time.

The deep learning based methods generally have higher accuracy. But the design can be complex and for training, it requires good number of iterations. So, It can be time consuming.

There are many intrusion detection systems available which have used traditional ML techniques in them. Most of them have used ML techniques like SVM, Random Forest, Decision tree and soon.

One of the papers has given conclusion that Random Forest performed considerably well compared to other models in findong the malicious packets, with high performance metrics and Mathews correlation coefficient rates when verified on NSL-KDD dataset.[8]

Some of the works have designed IDS which involves training and testing of models like Decision Tree, Gradient Boosting Tree, AdaBoost, Multilayer Perceptron, Long-Short Term Memory, and Gated Recurrent Unit for binary classification. The experimental results performed on UNSW-NB15 dataset indicated that Decision tree performed better in the experiment. [7]

In one of the other papers, when ML models are trained on, CIC-IDS-2017, UNSW-NB15, and CIC-IDS-2018 datasets, it resulted that Decision tree and Random Forest are performing well when tested with real time data. [6]

SVM has showed good performance in one of the other works related to IDS. [5]

Apart from the models which used traditional ML models, there are also models designed for Adhoc networks like one of the models divides the overall intrusion detection work into a four-level hierarchy. This will leads to a very energy-efficient structure. Each monitor only needs to monitor a few nodes in its area, so it doesn't need to consume much power. They presented a policy-based detection mechanism and intrusion response and a GSM cell concept for an intrusion response architecture.[4]

Other systems which include IDS using Genetic Algorithms. They employed the common KDD99 benchmark dataset to develop and assess the effectiveness of the system, and able to get a respectable detection rate. A chromosome's fitness was determined using the conventional deviation equation with distance. [2]

One of the other works includes using RST (Rough Set Theory) and SVM (Support Vector Machine) for intrusion detection. The RST-SVM method in framework resulted higher accuracy. [3]

There are also few models proposed by combining multiple models which included combining Twsvm, Decision tree and Hierarchical clustering. This has given 85% accuracy when verified on NSL-KDD data set.[1]

IV. TECHNOLOGY

Intrusion Detection Systems observe network traffic activities or analyse data generated by a host to detect any intrusion attempts that compromise security. These systems are the frontline defence against cybersecurity threats in the industry, raising alarms when an attack is detected. Machine learning is one of

the fastest growing technologies which mainly focuses on the data and many more different algorithms that imitate in the same way that human beings learn.

The Machine learning helps in improving the accuracy. It is one the branch of Artificial intelligence as well as the computer science. The machine learning allows the machine to learn automatically with the help of past information without programming the data clearly. Machine learning plays an important role in the upgradation of data science field. It uses many algorithms for building different mathematical models and it helps to predict the new output values based on the historic input values.

A decision tree is a supervised learning algorithm of the machine learning class. It performs well in general on problems dealing with classification and regression. Decision trees breaks the dataset into smaller subsets using recursive process. This effectively results in a tree-like model for making decisions. Each internal node is a test about an attribute, the outcome of the test is defined by each branch, and a class label is defined by each leaf node. The underlying theme is to make a model that detects the value for some target variable based on simple decision rules that are taken from the data features. Decision trees are intuitive in the interpretation and visualization of the structure but can easily overfit if not pruned properly.

A *Support Vector Machine* is one of the supervised ml algorithms which is used for classification models and sometimes in regression. It is primarily used in the domain of classification models. We represent every data item in an n-dimensional space, where n is the number of features. Here each feature is a value of a particular coordinate in the point.

Next, in classification, we find the hyperplane which can separates the data into the two classes at its best. Support vectors are the data points close to the hyperplane and define its position and orientation. The optimal separating hyperplane maximizes the margin between the support vectors of classes.

Twin Support Vector Machine (Twin SVM) is a ML algorithm derived from the regular Support Vector Machine (SVM). Unlike standard SVM, which constructs a single hyperplane to separate the classes, Twin SVM constructs two non-parallel hyperplanes, one for each class. These hyperplanes are closer to their respective classes and further from the other class, improving classification efficiency and accuracy.

Twin SVM offers significant advantages over traditional SVMs:

- Efficiency: Faster training times due to solving smaller quadratic programming problems.
- Robustness: Better handling of noisy data.
- Effectiveness: Improved performance on imbalanced datasets.

In a traditional SVM, given a set of training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$, where $\mathbf{x}_i \in \mathbb{R}^d$ are feature vectors and $y_i \in \{-1, 1\}$ are class labels, here the goal is to search for a hyperplane defined by: w.x=+b=0

such that it increases the distance between the hyperplane and support vectore(data points)

Twin support vector machine extends this idea by using two hyperplanes to better capture the separation between classes. The decision rule for Twin SVM can be formulated as follows:

For class $y = 1$:

$$\mathbf{w}_1 \cdot \mathbf{x} + b_1 \geq 1$$

For class $y = -1$:

$$\mathbf{w}_2 \cdot \mathbf{x} + b_2 \leq -1$$

Here, w1,b1 define the parameters of the first hyperplane and w2,b2 define the parameters of the second hyperplane.

The objective in Twin SVM is to simultaneously optimize the margins for both hyperplanes, subject to a penalty for misclassifications (similar to the soft-margin SVM formulation). The optimization problem can be written as:

$$\min_{\mathbf{w}_1, b_1, \mathbf{w}_2, b_2, \xi_i, \xi_i^*} \left(\frac{1}{2}(\|\mathbf{w}_1\|^2 + \|\mathbf{w}_2\|^2) + C\sum_{i=1}^{n}(\xi_i + \xi_i^*)\right)$$

subject to:

$$y_i(\mathbf{w}_1 \cdot \mathbf{x}_i + b_1) \geq 1 - \xi_i, \quad i = 1, \ldots, n$$
$$y_i(\mathbf{w}_2 \cdot \mathbf{x}_i + b_2) \leq -1 + \xi_i^*, \quad i = 1, \ldots, n$$
$$\xi_i, \xi_i^* \geq 0, \quad i = 1, \ldots, n$$

Here, regularization parameter is C which controls the trade-off between maximizing the margin and minimizing the classification errors. ξi and ξi* are variables that allow for misclassifications, similar to the soft-margin SVM.

To solve the Twin SVM optimization problem, various techniques such as quadratic programming or decomposition methods can be employed. The goal is to find the optimal parameters w1,b1, w2,b2 that maximize the margin between the two hyperplanes while correctly classifying the training data.

In Twin SVM, the optimization problem involves constructing two hyperplanes, one for each class, by solving two smaller quadratic programming problems. This reduces the computational complexity compared to standard SVM, which solves a larger single quadratic problem.

For class 1:

$$\min_{w_1,b_1} \|X_1 w_1 + e_1 b_1\|^2 + c_1 \|X_2 w_1 + e_2 b_1 + 1\|^2$$

For class 2:

$$\min_{w_2,b_2} \|X_2 w_2 + e_2 b_2\|^2 + c_2 \|X_1 w_2 + e_1 b_2 + 1\|^2$$

where X_1 and X_2 are the data matrices for the two classes, w_1 and w_2 are weight vectors, b_1 and b_2 are biases, and c_1 and c_2 are regularization parameters.

The TWSVM works in the following way

Constructing Two Hyperplanes

Twin SVM constructs two non-parallel hyperplanes, each optimized to be closer to one class and further from the other. This involves solving two quadratic programming problems, one for each class.

Solving Quadratic Programming Problems

The programming problems which are quadratic in Twin SVM are smaller and more efficient to solve than the single problem in standard SVM. This results in faster training times and reduced computational complexity.

Decision Function

The decision function in Twin SVM determines the class of a new data point which will be based on its distance from the two hyperplanes. The point is classified to the class of the hyperplane it is closer to.

Twin SVM is particularly effective for binary classification tasks, where it separates two classes with higher accuracy and efficiency.

While Twin SVM is mainly introduced for binary classification, it can be further updated to multiclass classification using different techniques like one-vs-one or one-vs-all approaches.By combining multiple TWSVMs, we can solve multi-class-classification problem.

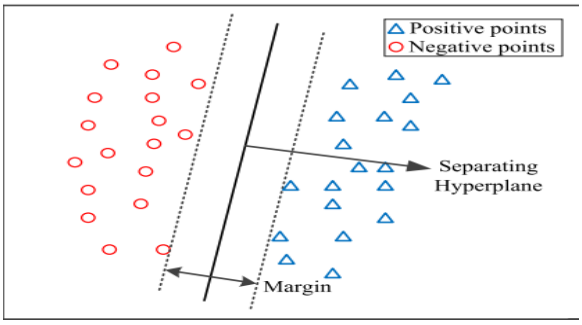

Fig. 1 Data classification in Support vector machine

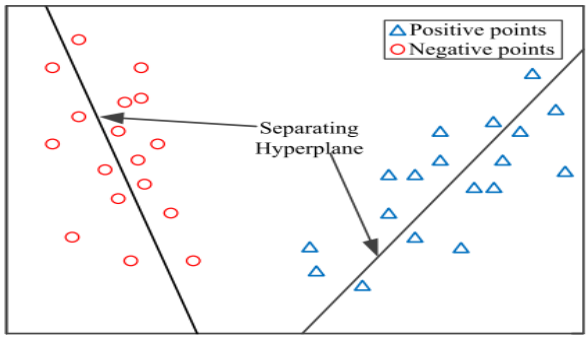

Fig. 2 Data classification in Twin SVM

V. SYSTEM ARCHITECTURE

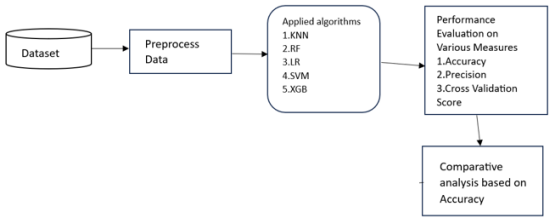

Fig.3 Architecture of the designed IDS

A. *Methodology*

As shown in the above architecture, the process can be depicted in 3 steps.

1. Data PreProcessing
2. Train and Evaluate models
3. Visualization

- Processing the imported data set is an important role in creating the realistic IDS.
- This processing includes multiple steps like, removing duplicate data, handling missing data, handling numeric data columns and some other erroneous data.
- We have used SMOTE to get the equal samples in all different classes.

- Once the processing done, this data is passed to the multiple ML models created.
- 2 models are proposed which is a combination of Decision tree/Random Forest, Hierarchical clustering and Twin SVM. This has resulted a good accuracy with less computation time.
- Both the combined models has given same accuracy.
- In the combined models, the following steps are implemented
    1. For the preprocessed data, firstly agglomerative hierarchical clustering is applied. This will lessen the error in the data classification during construction of decision tree.
    2. Once hierarchical clustering is done, the decision tree is built with the data which will effecively classify the data.
    3. Then TWSVMs are implemented in leaf nodes of decision tree which will again monitor network traffic before giving the output.
    4. As here the data detection is happening at 2 levels, i.e., Decision tree and TWSVM, this will improve th ecomputation efficiency and accuracy of the system.
- This model also got higher accuracy compared to previous projects related to the proposed system.
- Just not like the regular models which classifies either Attack or Not as output, Our proposed IDS system models will classify data according to the Outcome type instead of providing "Yes" or "No" Output.
- That is, it provides the output as "Normal" or if it's not, it gives out the attack name as well.
- Once the data sets are passed through all the ML models, we'll be visualising their outcomes in all different performance metrics.

B. Algorithm

1. Import the necessary libraries
2. Load the data set
3. Add columns to the data set
4.Encode the categorical variables, excluding outcome
5. Perform PCA to reduce dimensionality
6. Apply Agglomerative clustering and add cluster labels as features7. split the dataset
7. Check the min no. Of samples in any class
8. If min samples in class<1:
Apply SMOTE-ENN
9. Train a decision tree to segment the data set
10. Get the leaf node indices
11. Train Twin SVMs on each leaf node
12. Predict the result using decision tree and then the twin svm for the leaf node
13. Train additional classifiers(DT, Random forest, Gaussian Naive bayes, SVM, Gradient boosting)
14. Evaluate the results
15. visualization of results

V. EXPERIMENTAL RESULTS

In this section, proposed system was tested on benchamark data set named NSL-KDD. First the data set details are introduced then the results obtained by testing the data set are given.

A. Data set description

- NSL-KDD dataset which can be used by any one, and it was updated from KDD cup99 dataset which is taken from Kaggle.
- It has 125,973 records in training data and the test dataset comprises 18,794 records.
- The NSL-KDD dataset size is not high so it can be used for practical purpose directly without taking the random records.
- This data set has delivered good outcomes from various research works.
- It comprises 41 attributes (i.e., features).
- The dataset contains of several predictor variables and one target variable, Outcome.
- The data set size is 53Mb.
- "Outcome" is feature that we are going to predict, if it's normal or an attack.
- Anything other than "Normal" is attack in the dataset.

- The dataset consists of network connection records, each with 41 features. These features are divided into three categories:

- *Basic Features:* These include intrinsic attributes of individual TCP connections, such as service ,duration, protocol type, and flag.
- *Content Features:*These will include features derived from the payload of the original TCP packets, such as the number of failed login attempts.
- *Traffic Features:* These will include features found using a two second time window, like the number of connections to the same host in past two seconds.

- Every record is found as either normal or as an attack, where attacks falling into any of the four categories:
  - DoS(Denial of Service)
  - R2L (Remote to Local)
  - U2R (User to Root)
  - Probe

Category	Training data	Testing data
Normal	67343	9711
Probe	11656	1106
DOS	45927	5741
U2R	52	37
R2L	995	2199

Table. I Distribution of data in NSL-KDD data set

B. Results

The results for each of the trained models will be displayed in the following way.

The below are the reslts obtained from 2 combined models.

Combined Decision Tree and Twin SVM Accuracy: 0.9670
Combined Decision Tree and Twin SVM Classification Report

Attack	precision	recall	f1-score	support
ipsweep	0.86	0.67	0.75	9
neptune	1.00	1.00	1.00	101
nmap	1.00	1.00	1.00	6
normal	1.00	0.96	0.98	161
portsweep	0.88	0.88	0.88	8
satan	0.88	1.00	0.93	7
smurf	0.75	1.00	0.86	6
teardrop	1.00	1.00	1.00	3
warezclient	0.40	1.00	0.57	2
micro avg	0.97	0.97	0.97	303
macro avg	0.86	0.94	0.89	303
weighted avg	0.98	0.97	0.97	303

Fig.4 Result of Combined TWSVM and Decision tree

Combined Random Forest and Twin SVM Accuracy: 0.9670
Combined Random Forest and Twin SVM Classification Report:

Attack	precision	recall	f1-score	support
back	0.00	1.00	0.00	0
ipsweep	0.86	0.67	0.75	9
neptune	1.00	1.00	1.00	101
nmap	1.00	1.00	1.00	6
normal	1.00	0.96	0.98	161
pod	0.00	1.00	0.00	0
portsweep	0.88	0.88	0.88	8
satan	0.88	1.00	0.93	7
smurf	0.75	1.00	0.86	6
teardrop	1.00	1.00	1.00	3
warezclient	0.40	1.00	0.57	2
accuracy			0.97	303
macro avg	0.71	0.95	0.72	303
weighted avg	0.98	0.97	0.97	303

Fig. 5 Result of Combined TWSVM and Random forest

Accuracy:

Accuracy is defined as the ratio of the correctly predicted outcomes to the total outcomes.

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

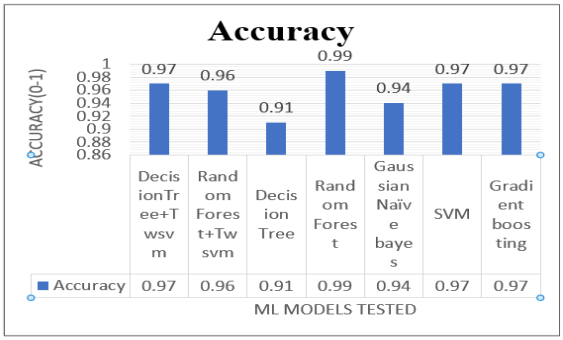

Fig.6 Comparison of accuracy over different models

Precision:

Precision is defined as the ratio of correctly predicted positive outcomes to the total predicted positives.

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

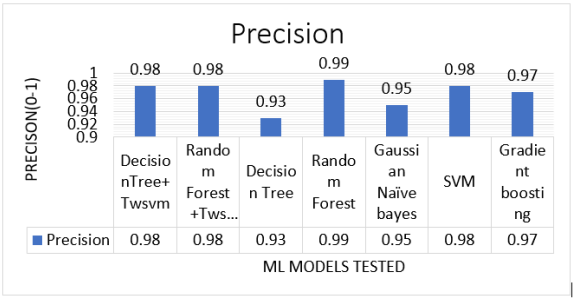

Fig.7 Comparison of precision over different models

Recall:

Recall is defined as the ratio of correctly predicted positive outcomes to all outcomes in the actual class.

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

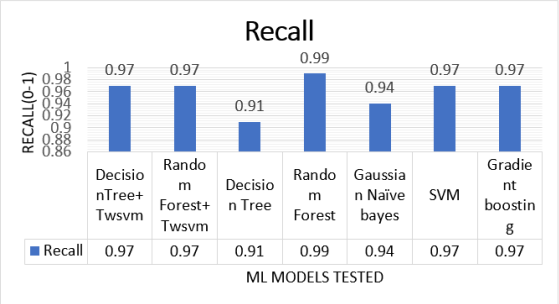

Fig. 8 Comparison of Recall over different models

F1 Score:

F1 Score is defined as the harmonic mean of Precision and Recall.

$$F1\ Score = 2\ X \frac{Precison\ X\ Recall}{Precison + Recall}$$

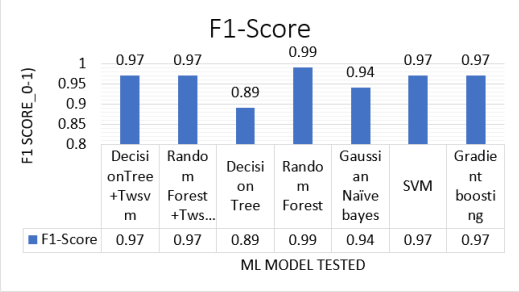

Fig. 9 Comparison of F1 Score over different models

Computation Time:

In machine learning (ML), computation time refers to the duration required to perform various tasks involved in the ML workflow.

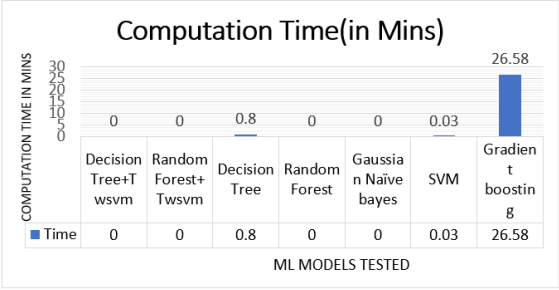

Fig. 10 Comparison of computation time over different models

C. Comparison with previously proposed models

Name of the model	Year of Implementation	Achieved Accuracy
BLSTM [11]	2020	79.40%
BAT [11]	2020	82.56%
BAT-MC [11]	2020	84.25%
BiLSTM [12]	2020	79.43%
CNN-BiLSTM [12]	2020	83.58%
AESMOTE [13]	2020	82.09%
DAE-DNN [14]	2021	83.33%
FL-DNN [15]	2021	75.13%
FL-CNN [15]	2021	75.26%
DE-ELM [16]	2022	80.15%
LSTM [17]	2022	79.00%
DNN [18]	2022	81.87%
CS-NN [19]	2022	85.56%
Deep-SARSA [20]	2023	84.36%
RNN-XGBoost [21]	2023	84.03%
LSTM-XGBoost [21]	2023	85.93%
GRU-XGBoost [21]	2023	85.65%
HC-DTTWSVM [1]	2023	85.95%
NIDS-HCTWSVM(Proposed System)	2024	97.02%

Table. II Comparison of proposed model with previosly proposed models

Compared to previous models, the current proposed model has given good accuracy as shown in above figure.

VI. CONCLUSION

In this project, we proposed an IDS which is a combination of Hierarchical clustering, decision tree/ Random Forest and Twin SVM. The main reson behind combining the models is, a single TWSVM cannot handle muti-class classification and even if we handle it, the accuracy is low. So, we used experimental approach and designed an IDS by combining models. Both the models(DT+TWSVM, RF+TWSVM) have given 0.97 accuracy using 80% of data when tested with NSL-KDD data set. Along with combined models, also verified the performance with traditional ML models like SVM, Decision tree Random Forest, Gradient boosting, and Gaussian naïve bayes. Among these, Random Forest, SVM and Gradient boosting are also giving the same accuracy as combined model (RF+TWSVM & DT+ TWSVM) but compared to combined model, they are having higher computation time. All the proposed model not only

tells us if the traffic is normal or not, it also detects the category of attack.

VII. FUTURE SCOPE

The obtained accuracy is by considering 80% of data. This accuracy can be improved more by increasing the size of the dataset.

Here we have tested the IDS by combining the 2 models(Decision tree and Random Forest) with TWSVM, this can be further improved by experimenting with other models.

Gradient boosting algorithm is taking long amount of time to compute, this computation time can be reduced by using advanced techniques without compromising on accuracy.

VIII. REFERENCES

[1] L. Zou, X. Luo, Y. Zhang, X. Yang and X. Wang, "HC-DTTSVM: A Network Intrusion Detection Method Based on Decision Tree Twin Support Vector Machine and Hierarchical Clustering," in IEEE Access, vol. 11, pp. 21404-21416, 2023, doi: 10.1109/ACCESS.2023.3251354.

[2] Hoque, Mohammad Sazzadul & Mukit, Md & Bikas, Md. Abu Naser. (2012). An Implementation of Intrusion Detection System Using Genetic Algorithm. International Journal of Network Security & Its Applications. 4. 109-120. 10.5121/ijnsa.2012.4208.

[3] Chen, Rung-Ching & Cheng, Kai-Fan & Hsieh, Chia-Fen. (2010). Using Rough Set and Support Vector Machine for Network Intrusion Detection. International Journal of Network Security & Its Applications. 1.

[4] Mamun, Mohammad & Kabir, A.. (2010). Hierarchical Design Based Intrusion Detection System For Wireless Ad Hoc Sensor Network. International journal of Network Security & Its Applications: IJNSA. 2. 102-117. 10.5121/ijnsa.2010.2307.

[5] U. S. Musa, M. Chhabra, A. Ali and M. Kaur, "Intrusion Detection System using Machine Learning Techniques: A Review," 2020 International Conference on Smart Electronics and Communication (ICOSEC), Trichy, India, 2020, pp. 149-155, doi: 10.1109/ICOSEC49089.2020.9215333.

[6] Talukder, Md. Alamin & Islam, Manowarul & Uddin, Md Ashraf & Hasan, Fida & Sharmin, Selina & Alyami, Salem & Moni, Mohammad Ali. (2024). Machine learning-based network intrusion detection for big and imbalanced data using oversampling, stacking feature embedding and feature extraction. Journal of Big Data. 11. 10.1186/s40537-024-00886-w.

[7] Abedin, Raisa & Waheed, Sajjad. (2022). Performance analysis of machine learning models for intrusion detection system using Gini Impurity-based Weighted Random Forest (GIWRF) feature selection technique. Cybersecurity. 5. 10.1186/s42400-021-00103-8.

[8] I. Sumaiya Thaseen, B. Poorva and P. S. Ushasree, "Network Intrusion Detection using Machine Learning Techniques," 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), Vellore, India, 2020, pp. 1-7, doi: 10.1109/ic-ETITE47903.2020.148.

[9] S. Ding, X. Zhao, J. Zhang, X. Zhang, and Y. Xue, ‘‘A review on multi-class TWSVM,’’ Artif. Intell. Rev., vol. 52, no. 2, pp. 775–801, Aug. 2019.

[10] A. A. Aburomman and M. B. I. Reaz, ‘‘A novel weighted support vector machines multiclass classifier based on differential evolution for intrusion detection systems,’’ Inf. Sci., vol. 414, pp. 225–246, Nov. 2017.

[11] T. Su, H. Sun, J. Zhu, S. Wang, and Y. Li, ‘‘BAT: Deep learning methods on network intrusion detection using NSL-KDD dataset,’’ IEEE Access, vol. 8, pp. 29575–29585, 2020.

[12] K. Jiang, W. Wang, A. Wang, and H. Wu, ‘‘Network intrusion detection combined hybrid sampling with deep hierarchical network,’’ IEEE Access, vol. 8, pp. 32464–32476, 2022.

[13] X. Ma and W. Shi, ‘‘AESMOTE: Adversarial reinforcement learning with SMOTE for anomaly detection,’’ IEEE Trans. Netw. Sci. Eng., vol. 8, no. 2, pp. 943–956, Apr. 2021.

[14] Y. N. Kunang, S. Nurmaini, D. Stiawan, and B. Y. Suprapto, ‘‘Attack classification of an intrusion detection system using deep learning and hyperparameter optimization,’’ J. Inf. Secur. Appl., vol. 58, May 2021, Art. no. 102804.

[15] M. Mulyanto, M. Faisal, S. W. Prakosa, and J.-S. Leu, ‘‘Effectiveness of focal loss for minority classification in network intrusion detection systems,’’ Symmetry, vol. 13, no. 1, p. 4, Dec. 2020.

[16] W. L. Al-Yaseen, A. K. Idrees, and F. H. Almasoudy, ‘‘Wrapper feature selection method based differential evolution and extreme learning machine for intrusion detection system,’’ Pattern Recognit., vol. 132, Dec. 2022, Art. no. 108912.

[17] E. Mushtaq, A. Zameer, M. Umer, and A. A. Abbasi, ‘‘A two-stage intrusion detection system with auto-encoder and LSTMs,’’ Appl. Soft Comput., vol. 121, May 2022, Art. no. 108768.
[18] S. P. Thirimanne, L. Jayawardana, L. Yasakethu, P. Liyanaarachchi, and C. Hewage, ‘‘Deep neural network based real-time intrusion detection system,’’ Social Netw. Comput. Sci., vol. 3, no. 2, p. 145, Mar. 2022.
[19] M. Rani, ‘‘Effective network intrusion detection by addressing class imbalance with deep neural networks multimedia tools and applications,’’ Multimedia Tools Appl., vol. 81, no. 6, pp. 8499–8518, Mar. 2022.
[20] S. Mohamed and R. Ejbali, ‘‘Deep SARSA-based reinforcement learning approach for anomaly network intrusion detection system,’’ Int. J. Inf. Secur., vol. 22, no. 1, pp. 235–247, Feb. 2023.
[21] S. M. Kasongo, ‘‘A deep learning technique for intrusion detection system using a recurrent neural networks based framework,’’ Comput. Commun., vol. 199, pp. 113–125, Feb. 2023.