Improving Accuracy of Intrusion Detection System using Stacked Ensemble Model and Voting Classifier

Redwan Ahmed Utsab
Department of Computer Science and Engineering
United International University
Dhaka, Bangladesh
rutsab222063@bscse.uiu.ac.bd

Rafi Abrar Kabir
Department of Computer Science and Engineering
United International University
Dhaka, Bangladesh
rkabir212012@bscse.uiu.ac.bd

Md. Koushik Ahmmed
Department of Computer Science and Engineering
United International University
Dhaka, Bangladesh
mahmmed212151@bscse.uiu.ac.bd

Amran Hossain
Department of Computer Science and Engineering
United International University
Dhaka, Bangladesh
amranhossain@example.com

***Abstract*—To address the ever-evolving landscape of cyber threats, establishing robust intrusion detection systems is critical for maintaining network security. This proposal outlines our plan to tackle the challenges of network intrusion detection by leveraging machine learning techniques. We intend to employ a range of classifiers, including Naive Bayes, Logistic Regression, Random Forest, and Voting Classifier, utilizing the comprehensive CICIDS2017 dataset. Our research will include an in-depth analysis of these classifiers, focusing on their performance in identifying network intrusions. While initial findings indicate promising results, the primary goal of this study is to examine the strengths and limitations of each algorithm within the context of intrusion detection. By conducting this research, we aim to emphasize the vital role of machine learning in strengthening network defenses and advancing our understanding of intrusion detection systems. The insights gained will pave the way for the development of more resilient and adaptive solutions to counter the ever-changing cyber threat landscape.**

***Index Terms*—Network Intrusion Detection, Machine Learning, Network Security, Ensemble Learning.**

I. Introduction

A network intrusion detection system (NIDS) is designed to detect unwanted or malicious traffic within a network, serving as a critical tool for safeguarding digital assets. With the increasing prevalence and sophistication of cyber-attacks, the demand for accurate and efficient intrusion detection systems has become more pressing. Recently, the field has seen a growing reliance on advanced techniques, particularly machine learning (ML) algorithms.

Thaseen et al. [13] demonstrated the successful application of various ML algorithms, achieving significant accuracy in detecting intrusions. Similarly, Ghani et al. [1] introduced an effective Feedforward Neural Network (FFNN) classifier for detecting network traffic anomalies, showcasing its capability across diverse datasets. Hidayat et al. [7] developed a hybrid feature selection technique that improved accuracy while reducing false positives and negatives. Meanwhile, Injadat et al. [8] proposed a multi-stage, optimized ML-based NIDS framework that outperformed previous approaches in intrusion detection accuracy.

This study builds on these efforts to strengthen digital network defenses against evolving threats, with a focus on improving the accuracy and effectiveness of intrusion detection systems. In summary, our key contributions include:

- Collection of a comprehensive dataset for intrusion detection.
- Comparative analysis with recent ML techniques.
- Enhancement of detection accuracy through ensemble learning.

This paper is structured as follows: Section II delves into the literature review and related works, Section III outlines our methodology, and Sections IV and V present the results, conclusion, and future work, respectively.

II. Literature Review

As cyber-attacks become increasingly diverse and sophisticated, developing reliable and effective intrusion detection systems is crucial for safeguarding sensitive network information. Numerous studies have addressed these challenges, aiming to enhance the accuracy of network intrusion detection.

Thaseen et al. [13] investigated the application of ML algorithms for accurate intrusion detection using a dataset captured via Wireshark, containing 1,130 instances and 24 attributes of various types. They evaluated algorithms such as Naïve Bayes, Logistic Regression, Random Forest, and Voting Classifier, achieving accuracy rates of 82.34%.

Ghani et al. [1] proposed a technique using a Feedforward Neural Network (FFNN) to detect network traffic anomalies

in the UNSW-NB15 and NSL-KDD datasets. With a reduced feature set, they achieved 91.29%.

Hidayat et al. [7] developed a hybrid feature selection method combining Pearson correlation coefficients and random forest models, achieving optimal accuracy with fewer false positives and negatives. Using ML and deep learning (DL) techniques such as Decision Tree, AdaBoost, K-Nearest Neighbors (KNN), Multilayer Perceptron (MLP), and Long Short-Term Memory (LSTM) on the TON IoT dataset, they achieved accuracies of 99.6%.

Injadat et al. [8] introduced a multi-stage, optimized ML-based intrusion detection framework. By leveraging feature selection techniques like information gain and correlation, along with hyperparameter optimization, their framework achieved 99%.

Rizvi et al. [10] utilized deep learning techniques to develop a flexible and efficient intrusion detection system. Their 1D-DCNN model was tested on CIC-IDS2017 and CSE-CIC-IDS2018 datasets, achieving accuracies of 99.7% and 99.98%, respectively. Future work includes comparing their model with other state-of-the-art deep learning approaches in terms of performance and computational cost.

Cao et al. [3] addressed dataset imbalance and feature redundancy using ADRDB and RFP algorithms, combined with CNN and GRU models for feature learning and an attention module to enhance performance. Evaluating their model on the NSL-KDD, UNSW-NB15, and CIC-IDS2017 datasets, they achieved accuracies of 99.69%, 86.25%, and 99.65%, respectively, with plans to focus on reducing time costs and improving detection.

Yang et al. [14] conducted a detailed literature survey of anomaly-based intrusion detection systems, analyzing 119 highly cited papers. They examined application domains, data preprocessing methods, detection techniques, evaluation metrics, and collaborative networks, identifying unresolved challenges and promising research directions.

Apruzzese et al. [2] emphasized cross-evaluation of ML-based NIDS using labeled data from multiple networks. Their XeNIDS framework demonstrated the potential and risks of cross-evaluations, highlighting contexts for analyzing diverse datasets.

Overall, these studies highlight the potential of machine learning-based classifiers to accurately detect network intrusions while managing large datasets, advancing the field of network security.

I. Methodology

A. Data Collection

1) **Dataset Overview**: The research utilized the CIC-IDS2017 dataset, which comprises multiple CSV files representing various cyberattack scenarios, such as DDoS attacks, port scans, web attacks, and infiltration. These scenarios were recorded across different days and timeframes.
2) **Data Preparation**: The individual files within the CIC-IDS2017 dataset were combined into a single DataFrame. During this process, missing and infinite values were meticulously addressed to maintain the dataset's integrity.

B. Exploratory Data Analysis (EDA)

1) **Feature Classification**: Dataset features were classified as either numerical or categorical, with particular attention given to identifying the label column for binary classification purposes.
2) **Label Transformation**: To enable binary classification, labels were re-mapped, assigning "BENIGN" as the normal class and grouping all other instances under the "INTRUSION" category.
3) **Data Examination**: The exploratory data analysis included a detailed examination of the distribution of categorical features and the handling of missing values in numerical features.

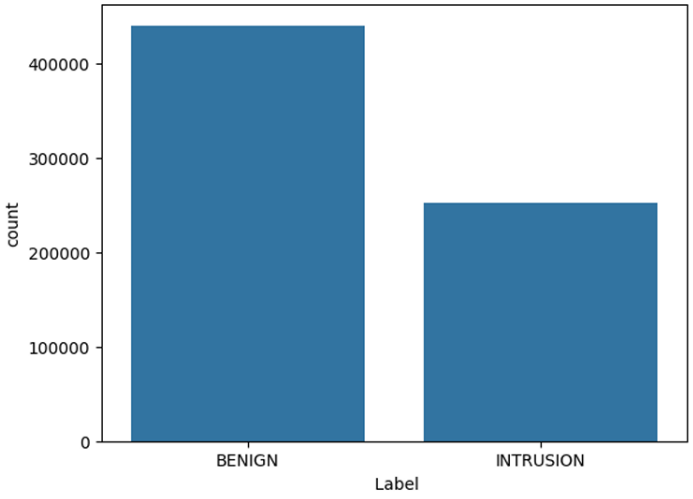

Fig. 1. Label Mapping from CIC IDS 2017 dataset

C. Data Transformation

1) **Feature Scaling**: Numerical features were normalized using Min-Max scaling to ensure consistent ranges across the dataset. The Min-Max Normalization formula is given by:

$$X_{\text{normalized}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{1}$$

Here, X represents the original data point, $X_{\text{normalized}}$ is the scaled value, $X_{\min}$ is the minimum value in the dataset, and $X_{\max}$ is the maximum value in the dataset.
2) **One-Hot Encoding**: Categorical labels were converted into numerical format using one-hot encoding, enabling machine learning models to process these features effectively.

D. Data Splitting

A crucial step was to divide the dataset into training and testing subsets. This ensured unbiased model evaluation and accurate performance assessment.

E. Model Implementation and Training

1) **Decision Tree Classifier**: A Decision Tree Classifier structures data into a hierarchical tree, making decisions based on feature values. It recursively partitions the data to create branches, where each leaf node represents either a class label or a regression outcome.
2) **Naive Bayes Classifier**: The Naive Bayes Classifier is a probabilistic model that utilizes Bayes' theorem to predict class labels. It assumes that features are independent given the class label. Despite this assumption, the model is computationally efficient and performs exceptionally well in text classification and other tasks.
3) **Logistic Regression**: Logistic Regression is a widely used model for binary classification that estimates the likelihood of an instance belonging to a specific class. It applies the logistic function to transform linear predictions into probabilities ranging between 0 and 1, making it particularly effective for probability estimation.
4) **Random Forest Classifier**: The Random Forest Classifier is an ensemble learning algorithm that constructs multiple decision trees during training. It determines the class by majority voting for classification tasks or calculates the mean prediction for regression. This approach enhances accuracy while minimizing overfitting.
5) **Voting Classifier**: The Voting Classifier aggregates predictions from multiple base classifiers through majority voting. By harnessing the collective knowledge of diverse models, it improves classification performance and robustness. This method is especially beneficial when individual models perform well in distinct areas or exhibit complementary strengths.
6) **Stacking**: Stacking is a meta-ensemble learning technique that integrates predictions from multiple base classifiers using a higher-level model. By optimizing the combination of model outputs, stacking enhances overall predictive accuracy and robustness. This method is well-suited for complex classification tasks where individual models may fail to capture all data patterns.

F. Model Evaluation and Interpretation

1) Confusion Matrix Analysis: Confusion matrices were used to assess model performance by analyzing true positives, true negatives, false positives, and false negatives.

2) Feature Importance: The contribution of different predictors was examined to determine their significance, particularly in tree-based models, to enhance model interpretability.

G. Performance Metrics and Comparison

1) Metric Evaluation: Essential performance metrics, such as accuracy, precision, recall, and F1 score, were systematically evaluated to compare different models effectively.

2) ROC and Precision-Recall Curves: Receiver Operating Characteristic (ROC) and Precision-Recall curves were generated to provide a detailed insight into model performance across different threshold settings.

H. Proposed Learning Model

1) Base Classifier Selection: A diverse set of classifiers, denoted as $h_1, h_2, ..., h_n$, was selected to capture various aspects of network traffic behavior. These classifiers were chosen for their complementary strengths in distinguishing between normal and malicious activities.

2) Training of Base Classifiers: Each classifier h_i was trained separately using feature vectors extracted from network packet data. The feature matrix X represents network packets, with rows corresponding to packets and columns representing features. During training, classifiers were optimized to predict the binary label y, classifying packets as either benign ($y = 0$) or malicious ($y = 1$).

3) Meta-Classifiers Construction: The predictions from the base classifiers on the training dataset are combined to serve as input for the meta-classifier. A soft voting approach is applied, where the final prediction $\hat{y}$ is determined by aggregating the weighted outputs of the base classifiers:

$$\hat{y} = \arg\max \left(\sum_{i=1}^{n} w_i \cdot h_i(X) \right)$$

where w_i denotes the weight assigned to each base classifier. These weights are determined based on classifier performance metrics or optimized through ensemble learning techniques such as meta-learning or genetic algorithms.

4) Model Optimization and Refinement: The stacked ensemble model undergoes an iterative process of optimization and refinement to enhance performance. This includes hyperparameter tuning, feature selection, and ensemble pruning techniques, ensuring improved generalization and robustness.

IV. Result Analysis

A. Analysis of the Proposed Model

Acronyms and abbreviations should be defined when first introduced in the text, even if they were previously defined in the abstract. However, commonly recognized abbreviations such as IEEE, SI, MKS, CGS, AC, DC, and RMS do not require definition. Additionally, abbreviations should be avoided in titles or section headings unless necessary.

B. Comparison of the Proposed Model with Other Machine Learning Algorithms

The **Naïve Bayes Classifier** exhibits relatively lower performance metrics in comparison to **Stacking Ensemble Learning**. While it achieves moderate accuracy, its F1 score, precision, and recall are significantly lower, indicating its limitations in capturing complex patterns within the dataset.

Logistic Regression delivers moderate performance in terms of accuracy and F1 score. However, its precision and recall are generally lower than those of the **Decision Tree**

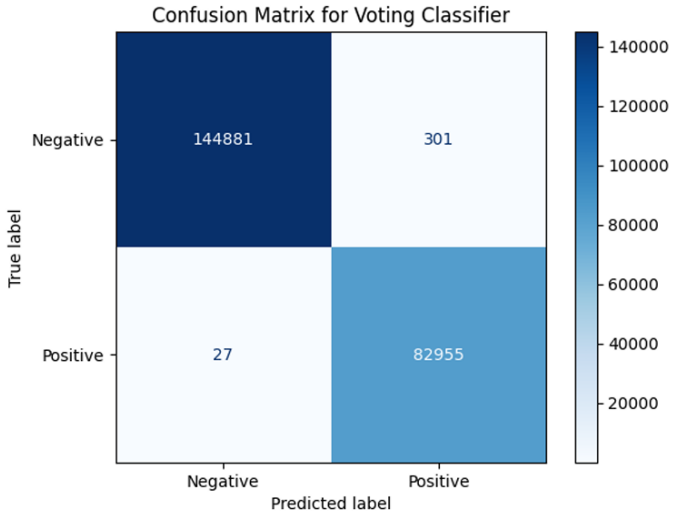

Fig. 2. Voting Classifier

Classifier, potentially impacting its effectiveness in detecting network intrusions.

Random Forest and Voting Classifiers exhibit strong performance, demonstrating high accuracy and F1 scores, which are nearly comparable to the **Decision Tree Classifier**. Their precision and recall rates are also notable, highlighting their robustness in identifying network intrusions. Our **stacked ensemble model** outperformed individual base classifiers across various evaluation metrics. The model achieved an **accuracy of 99.99%**, **precision of 99.94%**, **recall of 99.97%**, and an **F1 score of 99.95%** on the test dataset. These results emphasize its ability to effectively differentiate between normal and malicious network activities. Further **ROC (Receiver Operating**

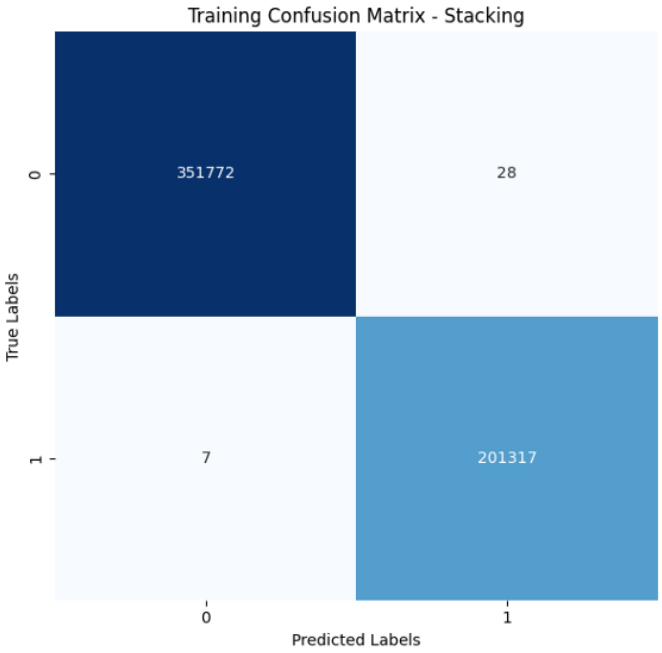

Fig. 3. Stacked learning on training

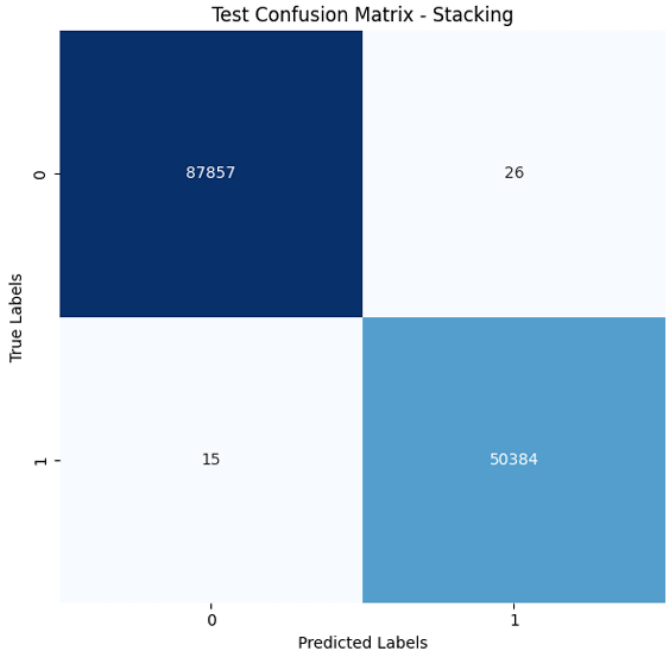

Fig. 4. Stacked learning on testing

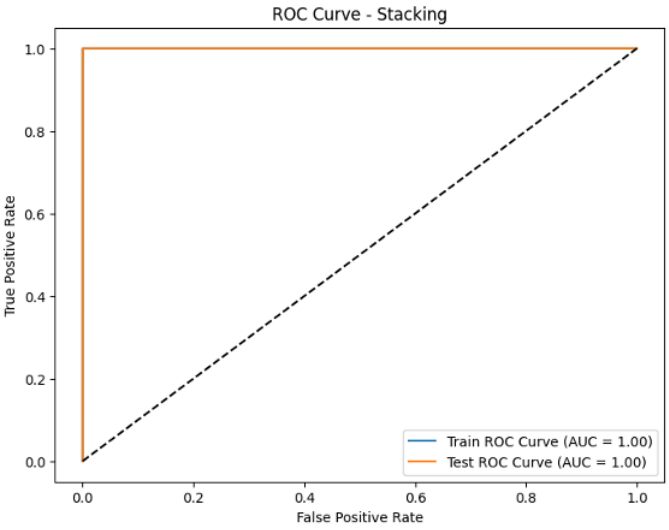

Fig. 5. ROC curve of stacking

Characteristic) analysis confirmed the model's reliability in distinguishing between benign and malicious traffic across different conditions. The **Area Under the Curve (AUC)** was measured at **1.00**, indicating exceptional discriminatory power with minimal false positive rates. This suggests that our model maintains a strong balance between **sensitivity and specificity** in network intrusion detection.

Conclusion and Future Work

This study highlights the effectiveness of utilizing a **stacking-based ensemble learning approach** with a **voting**

TABLE I
COMPARISON OF DIFFERENT MODELS

Model	Train Acc.	Test Acc.	F1 Score	Precision	Recall
Naïve Bayes	82.15%	82.34%	78.82%	69.90%	90.35%
Random Forest	99.96%	99.95%	99.93%	99.90%	99.96%
Logistic Regression	96.93%	96.95%	95.82%	95.60%	96.03%
Voting Classifier	99.89%	99.85%	99.80%	99.63%	99.96%
Proposed Model	99.99%	99.97%	99.95%	99.94%	99.97%

classifier as the base model to enhance **intrusion detection systems**. The proposed model demonstrates exceptional performance, achieving **high accuracy and F1 score**.

A detailed analysis provides insights into the **strengths and limitations** of various classifiers, including **Naïve Bayes, Random Forest, Decision Tree, K-Nearest Neighbors (KNN), and Logistic Regression**. The trade-off between accurately detecting intrusions and minimizing false positives is further examined using **precision and recall** metrics.

For future research, we suggest **exploring advanced ensemble techniques** to further enhance machine learning models' efficiency in **network security**. This work contributes to the advancement of **effective intrusion detection systems**, which are vital for proactively countering **emerging cybersecurity threats** and ensuring **robust network protection**.

REFERENCES

[1] H. Ghani, B. Virdee, and S. Salekzamankhani, "A deep learning approach for network intrusion detection using a small features vector," *Journal of Cybersecurity and Privacy*, vol. 3, no. 3, pp. 451–463, 2023.

[2] G. Apruzzese, L. Pajola, and M. Conti, "The cross-evaluation of machine learning-based network intrusion detection systems," *IEEE Transactions on Network and Service Management*, vol. 19, no. 4, pp. 5152–5169, 2022.

[3] B. Cao, C. Li, Y. Song, Y. Qin, and C. Chen, "Network intrusion detection model based on CNN and GRU," *Applied Sciences*, vol. 12, no. 9, p. 4184, 2022.

[4] B. Charbuty and A. Abdulazeez, "Classification based on decision tree algorithm for machine learning," *Journal of Applied Science and Technology Trends*, vol. 2, no. 1, pp. 20–28, 2021.

[5] A. Das, "Logistic regression," in *Encyclopedia of Quality of Life and Well-Being Research*, pp. 1–2, Springer, 2021.

[6] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," *IEEE Transl. J. Magn. Japan*, vol. 2, pp. 740–741, Aug. 1987. [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[7] I. Hidayat, M. Z. Ali, and A. Arshad, "Machine learning-based intrusion detection system: An experimental comparison," *Journal of Computational and Cognitive Engineering*, vol. 2, no. 2, pp. 88–97, 2023.

[8] M. Injadat, A. Moubayed, A. Bou Nassif, and A. Shami, "Multi-stage optimized machine learning framework for network intrusion detection," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1803–1816, 2020.

[9] A. Parmar, R. Katariya, and V. Patel, "A review on random forest: An ensemble classifier," in *International Conference on Intelligent Data Communication Technologies and Internet of Things (ICICI) 2018*, pp. 758–763, Springer, 2019.

[10] S. Rizvi, M. Scanlon, J. McGibney, and J. Sheppard, "Deep learning-based network intrusion detection system for resource-constrained environments," in *International Conference on Digital Forensics and Cyber Crime*, pp. 355–367, Springer, 2022.

[11] M. M. Saritas and A. Yasar, "Performance analysis of ANN and Naïve Bayes classification algorithm for data classification," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 7, no. 2, pp. 88–91, 2019.

[12] A. Shokrzade, M. Ramezani, F. A. Tab, and M. A. Mohammad, "A novel extreme learning machine-based KNN classification method for dealing with big data," *Expert Systems with Applications*, vol. 183, p. 115293, 2021.

[13] I. S. Thaseen, B. Poorva, and P. S. Ushasree, "Network intrusion detection using machine learning techniques," in *2020 International Conference on Emerging Trends in Information Technology and Engineering (IC-ETITE)*, pp. 1–7, IEEE, 2020.

[14] Z. Yang, X. Liu, T. Li, D. Wu, J. Wang, Y. Zhao, and H. Han, "A systematic literature review of methods and datasets for anomaly-based network intrusion detection," *Computers and Security*, vol. 116, p. 102675, 2022.