

Stacked Ensemble Learning for Effective Flow-Based Intrusion Detection System

Omar Abdulmunem Ibrahim Aldabbagh

Dr.omar.aldabbagh@uomosul.edu.iq

Networking Department, College of Computer Science and Mathematics, University of Mosul, Mosul, Iraq

Received: October 14th 2025 Received in revised form: January 27th 2026 Accepted February 10th 2026

ABSTRACT

As developing cyber threats complexity and sophistication, traditional Network Intrusion Detection Systems (NIDS) face difficulties in detecting attacks within an acceptable time. For these challenges, this paper presents a new flow-based NIDS framework utilizing the benefits of Advanced Machine Learning methods, specifically the heterogeneous stacking ensemble. That leads to enhancing the detection performance in different attack scenarios. The proposed system merges three different base learner algorithms: Naïve Bayes (NB), Random Forest (RF), and eXtreme Gradient Boosting (XGBoost) with two different stackers (meta-learners): J48 decision tree and Logistic Regression (LR). The proposed system uses the comprehensive benchmark CSE-CIC-IDS-2018 dataset for testing and evaluation. In order to reduce the dimension of the feature space and to improve the model efficiency, feature selection is used with the aid of Mutual Information (MI). The experimental results showed that the LR-based meta-learner outperforms J48, achieving superior results of 97.98%, 98.90%, 93.92%, and 96.34% for accuracy, precision, recall, and f1-score, respectively. These results represent significant enhancements over the individual base classifiers and traditional ensemble methods. This can be used to build modern, effective cybersecurity applications that combine diverse machine learning algorithms with strategic feature engineering.

Keywords:

Cyber security; Network Intrusion Detection; Machine learning; Ensemble learning; Stacking; Mutual Information; CSE-CIC-IDS-2018 dataset; Flow-based traffic.

This is an open access article under the CC BY 4.0 license (<http://creativecommons.org/licenses/by/4.0/>).

<https://jamh.uomosul.edu.iq/index.php/rengj>

Email: alrafidain_engjournal3@uomosul.edu.iq

1. INTRODUCTION

According to the great spread of connecting devices and the dependency on digital services, network security is becoming more vital. The development of cyber threats' complexity needs advanced technical defense. The Network Intrusion Detection System (NIDS) plays a big role in determining malicious activities and potential security breaches inside the network [1],[2]. Often, the traditional Network Intrusion Detection System (NIDS) performs deep packet inspection, which can be resource-intensive [3]; therefore, it is not suitable for high-speed networks or encrypted traffic. Flow-based NIDS can overcome this limitation as it can be considered as a practical alternative that analyzes network traffic metadata (e.g., source and destination IPs, ports, protocols, number of packets, and duration) instead of the packet payloads. Flow-based NIDS have vital characteristics, including privacy, scalability, and

performance, since there is no need to inspect sensitive data [4].

On the other hand, flow-based NIDS can use machine learning algorithms to distinguish between benign and malicious network traffic. The machine learning classifier depends entirely on its own assumptions and internal structure, which can decrease the performance. In contrast, ensemble learning combines the output of multiple classifiers to enhance the accuracy, generalization, and robustness by reducing bias and variance [5]. The next subsection presents an overview of Intrusion Detection Systems in cybersecurity and their challenges in detecting and defending against modern attacks. The ensemble learning methods, including bagging, boosting, and stacking, are traversed and focused on their relevance for enhancing the IDS robustness and efficiency. After that, the motivation and the contribution of the paper are demonstrated.

1.1. Intrusion detection systems (IDS)

Intrusion Detection System plays a vital role in protecting the systems and networks. It is an important component in the modern network infrastructure. Intrusion Detection Systems are designed for monitoring and analyzing the network traffic and system profiles to detect any malicious activity or violations of the system or network policies [6]. IDS is classified into two categories: Network Intrusion Detection System (NIDS) and Host Intrusion Detection System (HIDS). IDS protects against unauthorized access, denial of services (DoS), and many other malicious attacks [7].

While HIDS operates in detecting intrusions at the end devices, the NIDS operates on network traffic at the intermediate devices. Most traditional IDSs depend on signature-based detection. While traditional IDS is effective and fast in detecting already known intrusions, it fails in detecting new or zero-day attacks. This limitation motivates the adoption of anomaly-based recognition methods, especially those that are powered by ML methods, which enable such systems to learn normal activity and detect any deviations as potential threats. Among the above detection approaches, flow-based NIDSs have attracted attention since they provide scalability and efficiency in detecting malicious network activity via deep inspection of network traffic and extract metadata for robust detection. These traits enable IDSs to work in real time, at high speed, and in an encrypted environment, effectively [8].

1.2. Ensemble learning

Ensemble learning is an advanced ML technique that fuses several classification algorithms and produces a more accurate, robust, and reliable model than any standalone model. In the case of intrusion detection, ensemble techniques are valuable due to their capabilities to deal with diverse attack behaviors, hence minimizing the generalization error, and enhancing the ability to resist noise in traffic data. There are three widely adopted ensemble methods: Bagging, Boosting, and Stacking [9].

Bagging (or bootstrap aggregation) works by training multiple ML algorithms independently on randomly different subsets of the training dataset, then combining their outputs to produce a more stable model. Boosting, on the other hand, constructs models in a successive manner, where each new pretrained model focuses on correcting the errors arising from its predecessor model, thus enhancing the model to capture complex instances and reducing bias. The stacking ensemble method

introduces a new layer, where a set of base algorithms is trained in a parallel fashion, and their predictions are fused through a high-level learner layer (meta model) to produce a final prediction. This concept enables stacking to mitigate the complementary strength of diverse base learners

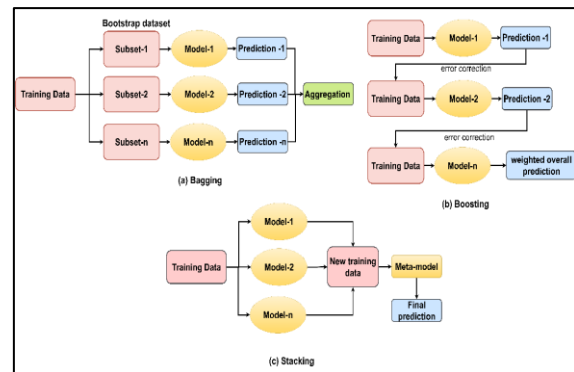


Fig. 1 Structure of basic ensemble methods

[10]. Figure 1 presents the structure of the ensemble methods, highlights their differences, and the work principles of each one to improve the classification model.

The growing and increasing severity of attacks imposes the limitation of traditional IDSs, especially in high-speed and evolving networks. Integrating ensemble methods with IDS can improve detection accuracy, provide adaptability, and achieve robustness for revealing both known and novel attacks. This paper presents a new robust flow-based NIDS by leveraging a heterogeneous stacking ensemble architecture to improve detection accuracy and scalability. Three diverse base learners are employed: NB, RF, and XGBoost. The proposed flow-based NIDS utilized the CSE-CIC-IDS-2018 [11], a comprehensive benchmark intrusion detection dataset, for training and evaluation purposes. The mutual information (MI) feature selection method is utilized to improve the efficiency of detection. The base learners RF, NB, and XGBoost were selected due to their complementary learning behavior, providing diverse base learners that are considered the concrete stone of the ensemble architecture since they reduce the correlated errors within the ensemble. In the meta-layer, the LR and J48 were utilized to investigate both linear and nonlinear combination strategies when fusing the base learners' output. This combination ensures a tradeoff stacking ensemble capable of revealing feature correlations, obviously when using the MI as feature selection subsets. Experimental results reveal that the proposed stacking ensemble achieves high performance across all evaluation metrics.

The rest of this paper is organized as follows: Section 2 highlights the related works, focusing on employing the stacking ensemble method on existing intrusion detection and the application of ensemble learning regarding cybersecurity. Section 3 presents the proposed methodology, including the preprocessing operation, feature selection, and the construction of the stacking ensemble. Experimental design, evaluation metrics, and results are outlined in Section 4. Finally, Section 5 concludes the paper and discusses the potential direction of future work.

2. RELATED WORK

Flow-based IDSs have been the subject of extensive literature since the emergence of severe cyberattacks. For improving detection accuracy and scalability, recent papers explored different ML algorithms and ensemble-based methods. In this section, a comprehensive review of previous critical works is presented, highlighting the development strategies, methodologies, and their achievements on benchmark datasets.

Alsaftar et al. [12] presented a new intrusion detection system method utilizing a special kind of ensemble called a multi-stacking ensemble (MSE). Their proposed system aims to improve the capabilities of attack prediction and overcome the limitations of using a single classifier when using a complex and imbalanced dataset. The method uses the LogitBoost algorithm with XGBRegressor as a feature selection method. Residual Network (ResNet) is employed for feature extraction. The paper uses the CICIDS2017 and UNSW-NB15 datasets for training and evaluating the proposed system. The work presented by Vullam et al [13] employed the stacking ensemble learning for improving the Intrusion Detection System. Three machine learning algorithms named: Decision Tree, Random Forest, and K-nearest neighbors are employed as a base learner, while Logistic Regression was used as the meta-learner. The paper achieved an accuracy of 96.16%, 97.95% for training and testing, respectively. The UNSW-NB15 dataset is used for evaluating the efficiency of the Intrusion Detection System.

In their work, Alalwany et al. [14] suggested a stacking ensemble method for an intrusion detection system (IDS) applied to Internet of Medical Things (IoMT). Traditional ML and deep learning DL approaches are used to improve detection accuracy. The IDS operates effectively in miscellaneous cyberattacks such as Denial of Services and Address Resolution Protocol spoofing. The method achieved an accuracy of 0.991 and 0.993 for binary and multi-

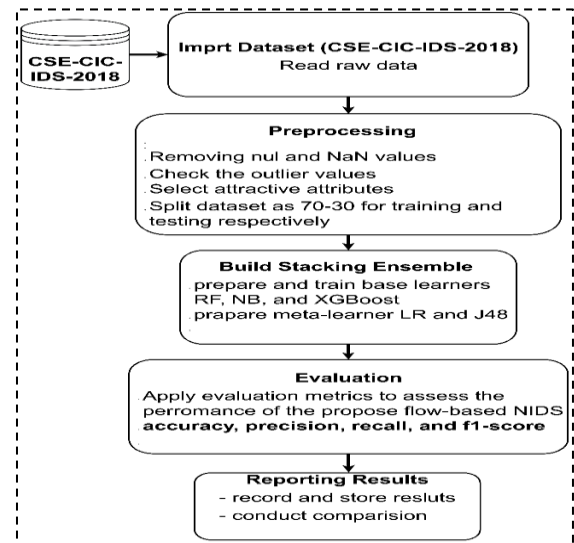


Fig. 2 The flow diagram of the proposed flow-based NIDS

class classification, respectively. The paper presented by Mills et al [15] proposed a method that uses both supervised and unsupervised learning techniques to form a stacking ensemble-based intrusion detection. The method uses three machine learning algorithms: a decision tree, a multiplayer perceptron, and a modified self-organizing map. The proposed ensemble method succeeds in minimizing the false alarms and improves the detection accuracy to 99.84% on the NSL-KDD and 99.84% on the CIC-DDOS2019 datasets.

Finally, Almomani et al [16] suggested an intrusion detection system that employs three ML algorithms, Decision Tree, K-Nearest-Neighbors, and Random Forest, to construct a stacking ensemble and combine via Logistic Regression as a meta-learner. The proposed IDS improves the classification performance on the UNSW-NB15 dataset and achieves an accuracy of 96.16% and 97.95% for training and testing, respectively.

3. METHODOLOGY

This section starts by presenting the methodology used for developing the proposed flow-based NIDS utilizing three heterogeneous base learners as stacking constituent classifiers: NB, RF, and XGBoost with two different meta learners, LR and J48 decision tree. All required components of the methodology are structured to facilitate the development of the proposed stacking flow-based NIDS. First, the CSE-CIC-IDS-2018 dataset is illustrated in terms of its features, records, and other information related to intrusion detection. Next, the preprocessing operations are conducted to achieve high-quality data to be used for training purposes and implement feature selection operation using the MI method, followed

by partitioning the dataset into 70-30 splits for training and testing, respectively. Then, the core of the methodology is demonstrated, which involves training the base learners on the same training dataset and the meta-learner on a new training dataset obtained, which reveals the stacking architecture. The final subsection of the methodology outlines the evaluation metrics used for assessing the proposed method. Figure 2 provides a flow diagram of the complete system workflow.

3.1. Data preprocessing

For this study, the CSE-CIC-IDS2018 dataset is utilized, which was developed by the Canadian Institute for Cybersecurity (CIC) and the Communications Security Establishment (CSE) [11]. This dataset is created in a simulation environment that can provide realistic network traffic and a wide range of modern attacks, such as Brute-force, DoS/DDoS, Web attacks, Infiltration, Botnet, and Heartbleed. This dataset includes approximately 16.23 million instances obtained from captured bidirectional flow in real time, utilizing CICFlowmeter, which also converts the obtained traffic into 80 important statistical and behavioral attributes. The capture of traffic reveals several categories, including content statistics data (as inter-arrival times and packet length), connection-based metrics (as duration and protocol type), and application-layer behaviors, etc. The dataset includes fifteen different attacks, falling under the general category of seven main attack categories. Table 1 shows the distribution of dataset attacks.

Table 1: The CSE-CIC-IDS2018 dataset attack types

Categories	Type
Brute Force	SSH-Bruteforce
	SSH-Patator
	FTP-Bruteforce
DoS	DoS Hulk
	DoS GoldenEye
	DoS Slowloris
	PortScan
	DoS Slowhttptest
DDoS	DDoS
Web Attacks	Web Attack – Brute Force
	Web Attack – XSS
	Web Attack – SQL Injection
Infiltration	Infiltration
Heartbleed	Heartbleed
Normal Traffic	Benign

The dataset traffic is organized in fourteen CSV files, and the obtained traffic is simulated by using 30 servers along with 400 PCs representing the victim machines, and 50 terminals act as the attacking infrastructure. It is worth mentioning that the dataset suffers from significant class imbalance, where certain attack classes are

heavily represented, while others contain only a small fraction of patterns.

Each of the CSV files includes a specific attack along with benign records. The total number of instances is 8,065,294, which reflects all types of attacks and normal traffic. For our proposed work, all attack records will be relabeled as “ATTACK”; thus, the produced dataset includes just two classes with 5778971 and 2286323 instances for Benign and ATTACK classes, respectively.

Several crucial preprocessing steps were applied to the newly obtained dataset, including removing missing values, checking for NaN and Null entries, followed by the normalization process. For this work, scalar or (Min-Max) normalization is applied to the CSE-CIC-IDS2018 dataset. This method converts the numeric values into a fixed distribution. The range normally is between 0 and 1. This can help to ensure that all attributes are equally represented in the training stage. The scalar normalization equation is as follows [17]:

$$\bar{N} = \frac{N - N_{\min}}{N_{\max} - N_{\min}} \quad (1)$$

Where $\bar{N}$ represents the normalized values in the range between [0,1], $N_{\max}$ is the maximum value of the attribute, $N_{\min}$ is the minimum value of the attribute, and N is the original value.

The feature selection is the next preprocessing step. For this work, Mutual Information (MI) is used to choose the attractive and effective features in order to enhance the performance and the efficiency of the proposed flow-based NIDS. MI measures the dependency between the features and the class label, which can extract the most effective features in the classification process. MI can reduce the mathematical complexity, thereby increasing the training process speed and reducing the memory consumption. Besides that, the feature selection mitigates against overfitting [18].

In the proposed work, MI is executed in two separate rounds: first, the most effective 25 features are extracted from the original 80 features. The proposed stacking flow-based NIDS system is trained and evaluated on the new reduced feature set. Second, only the top 15 effective features are selected. The stacking model also enters training and evaluation using this reduced feature set.

3.2. Stacking ensemble architecture

A two-layer stacking architecture is employed in the proposed flow-based NIDS for enhancing the classification performance and increasing model robustness against diverse threats. Our proposed architecture includes

heterogeneous base learners with a meta-layer learner to form the stacking model. Figure 3 provides a detailed view of each distinct stage and demonstrates the overall pipeline of the proposed system. The figure starts with importing raw data (CSE-CIC-IDS-2018), then the preprocessing and feature selection stage, which contains cleaning, outliers, normalization, and then the feature selection process. The experimental machine learning pipeline is the next stage, which splits into two main paths before reaching the final comparative evaluation.

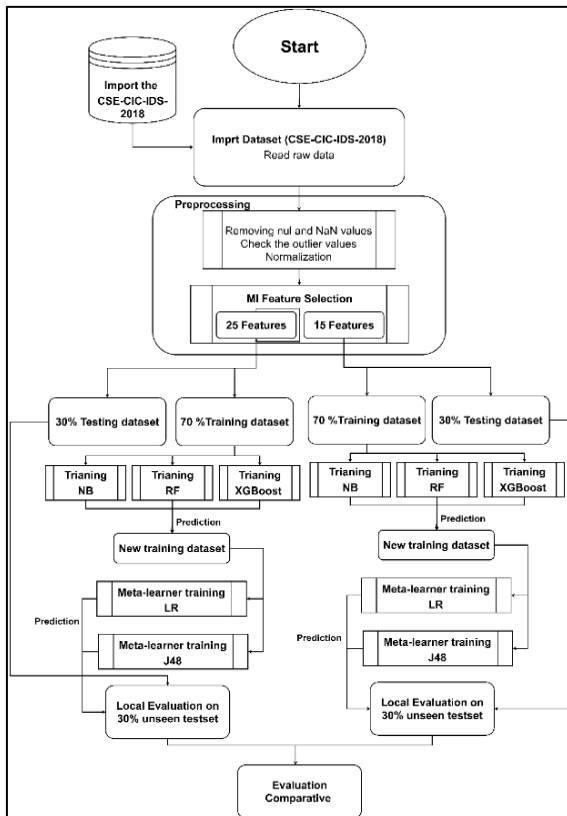


Fig. 3 Flowchart pipeline of the proposed flow-based NIDS

3.2.1. Naïve Bayse (NB)

The Naïve Bayes algorithm depends on probabilistic theory [19]. It relies on the strong assumption that it imposes independent conditional features. NB can be applied for classification and outlier detection since it can handle large data in an efficient manner. The equation below describes Bayes' theorem [20]:

$$P(X|C) = P(C|X).P(X)|P(C) \quad (2)$$

Where $P(X|C)$ represents the probability of getting class C when the feature is presented. The previous probability of class C is determined by $P(X)$; $P(C|X)$ refers to the likelihood of instance X given class C . $P(C)$ stands for the predictor's prior probability. Notably, NB is considered an excellent

classifier, represented by speed and low computation cost, even when dealing with large or imbalanced datasets.

3.2.2. Random Forest

Leo Breiman [21], who belongs to the University of California, is the discoverer of the random forest algorithm. The main idea behind the algorithm depends on generating many independent decision trees. Each sub-tree produces its own prediction, and the final winner output is aggregated by a voting method.

3.2.3. XGBoost

Extreme Gradient Boosting (XGBoost) is a powerful, effective, and scalable algorithm. It depends on the gradient boosting methodology. XGBoost is used in classification and regression tasks. It is also adaptable when applied to these tasks. Since the algorithm uses an advanced approach in constructing a scalable tree, it gains the ability to handle missing data and mitigate against overfitting. Moreover, the XGBoost could be tuned to work in a parallel processing environment, which is a crucial feature when dealing with a huge dataset [22].

3.2.4. Logistic Regression (LR)

Logistic Regression (LR) is an attractive statistical ML algorithm. LR is utilized for binary and multiclass tasks. It is mainly employed for regression, as its name implies, and it works by converting the normalized input patterns into ranges between $[0,1]$ as a vector. The newly produced vector data will be used to figure out the relationship between attributes and log odds. The LR equation is presented in the equation below [23]:

$$Ln(P1 - P) = \beta_0 + \beta_1x_1 + \beta_2x_2 + \dots + \beta_kx_k \quad (3)$$

Where Ln refers to a regression function, P represents the probability, X denotes the risk factor, and β is a constant of 1.

3.2.5. J48 decision tree

The J48 is a decision tree used for classification tasks, and it is an extension of the old C4.5 tree proposed by Quinlan in 1993 [24]. J48 depends on the divide and conquer idea, in which the tree is extensively split on the dataset, and each produced leaf node acts as a classification rule. Nevertheless, the performance of the tree decreases when dealing with a multiclass dataset, whereas it performs well when implemented on a binary classification task [25].

The procedure of the proposed system is outlined in Algorithm 1, which describes the steps of constructing and implementing the proposed

flow-based NIDS based on stacking ensemble through combining RF, NB, and XGBoost with LR and J48 as meta-learners.

3.3. Evaluation Metrics

The assessment of the proposed flow-based NIDS is conducted based on four key evaluation metrics: accuracy, precision, recall, and F1-score. These metrics are derived from a special matrix called the Confusion Matrix (CM), which includes numerical values regarding the prediction results in terms of False Negative (FN), False Positive (FP), True Negative (TN), and True Positive (TP). The CM is demonstrated in Table 2.

```

Algorithm 1: Flow-Based NIDS Using Stacking Ensemble
Input: D ← CSE-CIC-IDS2018 dataset (CSV)
Output: Comparison tables, ROC curves, confusion matrices
Begin
  Step 1: Load dataset D
  Step 2: Preprocess D
    Remove nulls, NaN values, duplicates
    Encode categorical attributes
    Encode class labels
  Step 3: Feature Selection using Mutual Information (MI)
    F25 ← Select top 25 features from D
    F15 ← Select top 15 features from D
  Step 4: Normalize selected features using Min-Max Scaling
  Step 5: Split dataset into Train (70%) and Test (30%)
  For each feature set F ∈ {F25, F15} do:
    Step 6: Train base learners on F
      Model_RF ← Train Random Forest
      Model_NB ← Train Naïve Bayes
      Model_XGB ← Train XGBoost
    Step 7: Generate meta-features
    Step 8: Train meta-learners
      Meta_LR ← Logistic Regression
      Meta_J48 ← J48
    Step 9: Evaluate stacked model on Test set based on Accuracy-Precision-Recall-F1-score, plot ROC-AUC
  End For
  Step 10: Compare performance of stacked models with individual base learners
End

```

Table 2: Demonstration of CM

		Predicted class	
		Negative class	Positive class
Actual class	Negative class	True negative (TN)	False positive (FP)
	Positive class	False negative (FN)	True positive (TP)

Based on information provided in CM, the four-evaluation metrics are determined as the equations below [8]:

$$Accuracy = \frac{(TP+TN)}{(TP+TN+FP+FN)} \quad (4)$$

$$Recall = \frac{TP}{(TP+FN)} \quad (5)$$

$$Precision = \frac{TP}{(TP+FP)} \quad (6)$$

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (7)$$

4. EXPERIMENTAL DESIGN AND RESULTS

In this section, the experimental procedure used to assess the performance of the proposed flow-based NIDS is demonstrated. Then, the results obtained from implementing the proposed system are discussed. For this work, all experiments were implemented according to the following specifications: a personal notebook equipped with an Intel(R) Core (TM) i5-2410M CPU @ 2.30 GHz and 8 GB of RAM. Python 3.10 programming language, and Jupyter notebooks, operated through the Anaconda distribution, are utilized for implementing the proposed system. These configurations were sufficient to handle the dataset, conduct the training for the base and meta learners, and evaluate the proposed stacking flow-based NIDS.

Results in Table 3 provide a performance comparison of the proposed flow-based NIDS utilizing the stacking ensemble learning with diverse configurations of the constituent base classifiers and feature subsets. More precisely, two stacker models (meta-learners): LR and J48 were assessed using the top 25 and 15 features obtained by implementing the MI feature selection method.

Table 3: Evaluation metrics comparison of the proposed flow-based NIDS

	Top (25) feature by MI		Top (15) feature by MI	
	LR	J48	LR	J48
Accuracy	0.9798	0.9642	0.9786	0.9590
Precision	0.9890	0.9328	0.9875	0.9167
Recall	0.9392	0.9415	0.9365	0.9407
F1-score	0.9634	0.9372	0.9613	0.9286

For Table 3, it is clear that the results obtained from employing LR with the top 25 attributes recorded superior performance for accuracy, precision, and F1-score; the recall value is slightly higher in the J48 model. The degradation in the recall in LR is due to the inherent balance between the recall and precision. Since the LR is considered a linear algorithm, it tends to provide more conservative decision boundaries, which in turn reduces the false positives, hence increases the precision and f1-score. On the other hand, J48 is considered more sensitive to minority classes, which improves the recall at the cost of more false positives, leading to low precision and overall performance.

From the previous results, it is suggested that using a richer attribute set will significantly aid in the generalization of the mode and minimize the FP, because the ensemble captures more informative features.

The J48, as a stacker with the same 25 features, showed lower performance, with an F1-

score of 93.72%, although exposing a higher recall of 94.15%, reflecting the sensitivity in capturing actual threats. However, reducing the feature space to 15 features caused a notable degradation in the precision and F1-score for both stacker models, especially for the J48 meta-learner.

The LR model with 15 attributes remains stable in its performance with an F1-score recorded of 96.13%, while J48 shows a lower value of the same metric of 92.86%, indicating that J48's base learner could be more affected by minimizing feature space. Broadly speaking, both configurations of 15 features showed high recall, meaning that even with a low number of features, the models maintained the ability to identify the attack patterns. Figure 4 demonstrates the findings supported by CMs, which illustrate the classification results for each configuration.

Figure 4 reveals that LR significantly shows fewer FP and FN compared to J48 meta-learner, specifically when implementing the full 25 attribute set. This highlights the advantages of LR as a stacker in balancing the generalization and accuracy of the imbalanced datasets, like intrusion detection.

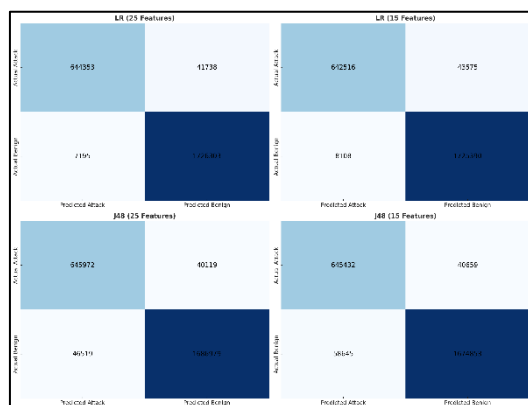


Fig. 4: CM comparison of the stacking ensemble with 25 and 15 feature sets applied with J48 and LR meta-learners

The Receiver Operating Characteristic (ROC) curve is a vital graphical evaluation metric used to assess the performance of binary

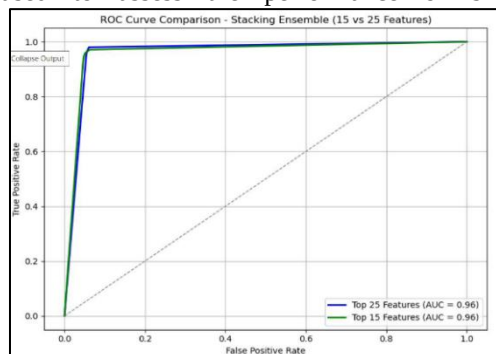


Fig. 5: ROC comparison of J48 meta-learner when running on 15 and 25 feature sets learners

classification tasks. ROC plots the Recall (also called True Positive Rate TPR) against the False Positive Rate FPR at different threshold values. This graphical metric demonstrates the trade-off between specificity and sensitivity, and highlights the behavior of models in discriminating between classes. A model that shows a curve closer to the left top corner means significantly better accuracy. Figure 5 shows the ROC comparison of the J48 meta-learner when running on 15 and 25 feature sets.

5. CONCLUSION AND FUTURE WORK

This paper proposed a new flow-based NIDS that leverages a stacking ensemble framework to detect and classify network attacks efficiently, utilizing the CSE-CIC-IDS2018, a benchmark dataset. Multiple base learners were employed, named: RF, NB, and XGBoost, with two meta-learners, J48 and LR. For increasing learning efficiency and minimizing computational complexity, Mutual Information was chosen as a feature selection method, thereby reducing the overall features from 80 to 25 and subsequently to 15 subsets. In both rounds, the obtained results demonstrated that the proposed stacking ensemble system shows high efficiency and exposed superior evaluation metrics, especially when implementing 25 feature subsets with LR. The stacking ensemble confirmed from empirical metrics its ability not only to strengthen the detection behavior but also to show high levels of reliability when dealing with different attack types. Moreover, the experimental results reflect that even with reduced feature space, the stacking ensemble continues to produce strong predictive accuracy.

In the near future, the proposed flow-based NIDS will be extended to detect threats in a real-time scenario by integrating the Flowmeter tool developed by the Canadian Centre for Cyber Security. Moreover, an advanced deep learning model will be explored, transformer models like BERK. We will address the limitation of class imbalance to improve the detection performance, particularly for minority malicious classes.

REFERENCES

- [1] S. M. Kasongo and Y. Sun, "A deep learning method with wrapper based feature extraction for wireless intrusion detection system," *Computers & Security*, vol. 92, p. 101752, 2020, doi: 10.1016/j.cose.2020.101752.
- [2] R. Lazzarini, H. Tianfield, and V. Charissis, "A stacking ensemble of deep learning models for IoT intrusion detection," *Knowledge-Based Systems*, vol. 279, p. 110941, Nov. 2023, doi: 10.1016/j.knosys.2023.110941.
- [3] W. Song *et al.*, "A software deep packet inspection system for network traffic analysis and anomaly

- detection," *Sensors*, vol. 20, no. 6, p. 1637, Mar. 2020, doi: 10.3390/s20061637.
- [4] L. Santos, R. Gonçalves, C. Rabadao, and J. Martins, "A flow-based intrusion detection framework for internet of things networks," *Cluster Computing*, vol. 26, no. 1, pp. 37–57, Feb. 2023 doi: 10.1007/s10586-021-03238-y.
 - [5] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications surveys & tutorials*, vol. 18, no. 2, pp. 1153–1176, 2015, doi: 10.1109/COMST.2015.2494502.
 - [6] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, "Deep learning approach for intelligent intrusion detection system," *IEEE access*, vol. 7, pp. 41525–41550, 2019, doi: 10.1109/ACCESS.2019.2895334.
 - [7] S. Muneer, U. Farooq, A. Athar, M. Ahsan Raza, T. M. Ghazal, and S. Sakib, "A critical review of artificial intelligence based approaches in intrusion detection: A comprehensive analysis," *Journal of Engineering*, vol. 2024, no. 1, p. 3909173, 2024, doi: 10.1155/2024/3909173.
 - [8] N. H. Al-A'araji, S. O. Al-Mamory, and A. H. Al-Shakarchi, "Classification and clustering based ensemble techniques for intrusion detection systems: A survey," in *Journal of Physics: Conference Series*, IOP Publishing, 2021, p. 012106, doi: 10.1088/1742-6596/1818/1/012106.
 - [9] L. Rokach, "Ensemble methods for classifiers," in *Data mining and knowledge discovery handbook*, Springer, 2005, pp. 957–980, doi: 10.1007/0-387-25465-X_45.
 - [10] X. Wu and J. Wang, "Application of bagging, boosting and stacking ensemble and easy ensemble methods for landslide susceptibility mapping in the three gorges reservoir area of China," *International Journal of Environmental Research and Public Health*, vol. 20, no. 6, p. 4977, Mar. 2023, doi: 10.3390/ijerph20064977.
 - [11] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," *ICISSp*, vol. 1, No. 2018, pp. 108–116, 2018, doi: 10.5220/0006639801080116.
 - [12] A. M. Alsaffar, M. Nouri-Baygi, and H. Zolbanin, "Enhancing intrusion detection systems with dimensionality reduction and multi-stacking ensemble techniques," *Algorithms*, vol. 17, no. 12, p. 550, 2024, doi: 10.3390/a17120550.
 - [13] N. Vullam, D. Roja, N. Rao, S. S. Vellela, L. R. Vuyyuru, and K. K. Kumar, "An Enhancing Network Security: A Stacked Ensemble Intrusion Detection System for Effective Threat Mitigation," in *2023 3rd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA)*, IEEE, 2023, pp. 1314–1321, doi: 10.1109/ICIMIA60377.2023.10426091.
 - [14] E. Alalwany, B. Alsharif, Y. Alotaibi, A. Alfahaid, I. Mahgoub, and M. Ilyas, "Stacking ensemble deep learning for real-time intrusion detection in IoMT environments," *Sensors*, vol. 25, no. 3, p. 624, 2025, doi: 10.3390/s25030624.
 - [15] G. A. Mills, D. K. Acquah, and R. A. Sowah, "Network intrusion detection and prevention system using hybrid machine learning with supervised ensemble stacking model," *Journal of Computer Networks and Communications*, vol. 2024, no. 1, p. 5775671, May 2024, doi: 10.1155/2024/5775671.
 - [16] A. Almomani *et al.*, "Ensemble-based approach for efficient intrusion detection in network traffic," *Intelligent Automation & Soft Computing*, vol. 37, no. 2, pp. 2499–2517, 2023, doi: 10.32604/iasc.2023.039687.
 - [17] M. Alfath *et al.*, "Detecting DDoS Attacks in Network Traffic Based on Supervised Machine Learning Techniques," *Al-Noor Journal for Information Technology and Cybersecurity*, vol. 1, no. 0, pp. 11–21, 2024, doi: 10.69513/jnfit.v1.i0.a2
 - [18] J. Cheng, J. Sun, K. Yao, M. Xu, and Y. Cao, "A variable selection method based on mutual information and variance inflation factor," *Spectrochimica Acta Part A: Molecular and Biomolecular Spectroscopy*, vol. 268, p. 120652, 2022, doi: 10.1016/j.saa.2021.120652.
 - [19] N. H. Al-A'araji, S. O. Al-Mamory, and A. H. Al-Shakarchi, "Constructing decision rules from naive bayes model for robust and low complexity classification," *International Journal of Advances in Intelligent Informatics*, vol. 7, no. 1, pp. 76–88, 2021, doi: 10.26555/ijain.v7i1.578.
 - [20] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian network classifiers," *Machine learning*, vol. 29, no. 2, pp. 131–163, 1997, doi: 10.1023/A:1007465528199.
 - [21] M. C. P. Saheb, M. S. Yadav, S. Babu, J. J. Pujari, and J. B. Maddala, "A review of DDoS evaluation dataset: CICDDoS2019 dataset," in *International Conference on Energy Systems, Drives and Automations*, pp. 389–397, Springer, 2021, doi: 10.1007/978-981-99-3691-5_34.
 - [22] S. S. Dhaliwal, A.-A. Nahid, and R. Abbas, "Effective intrusion detection system using XGBoost," *Information*, vol. 9, no. 7, p. 149, 2018, doi: 10.3390/info9070149.
 - [23] N. S. I. Husin, S. A. Mostafa, M. M. Jaber, S. S. Gunasekaran, A. H. Al-Shakarchi, and N. F. Abdulsattar, "Machine Learning Regression Approach for Estimating Energy Consumption of Appliances in Smart Home," in *2023 Al-Sadiq International Conference on Communication and Information Technology (AICCIT)*, IEEE, pp. 229–233, Al-Muthana, Iraq 2023, doi: 10.1109/AICCIT57614.2023.10217991.
 - [24] J. R. Quinlan, "C4.5: Programs for Machine Learning". Elsevier, 2014. ISBN: 978-1-55860-238-0. [Online]. Available: https://books.google.iq/books/about/C4_5.html?id=b3ujBQAAQBAJ
 - [25] S. Aljawarneh, M. B. Yassein, and M. Aljundi, "An enhanced J48 classification algorithm for the anomaly intrusion detection systems," *Cluster Computing*, vol. 22, no. Suppl 5, pp. 10549–10565, 2019, doi: 10.1007/s10586-017-1109-8.