ISA Journal of Multidisciplinary (ISAJM)

Homepage: https://isapublisher.com/isajm/
Email: office.isapublisher@gmail.com

Volume 2, Issue 5, Sept-Oct, 2025 ISSN: 3049-1851

An ensemble Models to detect intrusion using CIC-IDS-2017 and CIC-IDS-2018 as benchmarks

Olorunyomi Segun Omotayo, Oguntimilehin Abiodun, Josephine Olamatanmi Mebawondu & Olanike Christianah Akinduyite

Department of Computer Science, Afe Babalola University, Ado-Ekiti

Received: 18.08.2025 | **Accepted:** 05.09.2025 | **Published:** 07.09.2025

***Corresponding Author:** *Olorunyomi Segun Omotayo*

DOI: *10.5281/zenodo.17073568*

Abstract

Original Research Articles

This paper is based on an in-depth discussion of a hybrid ensemble-based intrusion detection platform, using two popular benchmark datasets CIC-IDS-2017 and CIC-IDS-2018. The model proposed is a combination of six different machine learning algorithms, Logistic Regression, Naive Bayes, K-Nearest Neighbors, Support Vector Machines, Decision Trees and Random Forests, and a two-layer ensemble model that combines stacking and majority voting methods. The goal of this hybridization is to leverage the personal strengths of each classifier and alleviate some of their weaknesses by making decisions together. The effectiveness of the model was measured using the key performance measures of accuracy, precision, recall and F1-score. In experiments, it can be shown that the ensemble model surpasses each individual classifier in both datasets in terms of detection rate and false positive rate. Indicatively, for example the hybrid model achieved average accuracy of over 96, whereas one algorithm achieved a sub-90 average. The results reveal the robustness and adaptability of the ensemble learning to the different patterns of attacker and traffic behavior. Another testimony that the model can be applied to the cybersecurity sphere is the enormous generalization to datasets. In order to improve the performance of the system and the working efficiency in the dynamic network environment, further experimentation on how the features of deep learning, real-time traffic modeling, and periodic thresholding can be incorporated into the system will be addressed in the future.

Keywords: *Intrusion detection, ensemble learning, machine learning, CIC-IDS-2017, CIC-IDS-2018, cybersecurity, classification performance.*

1. Intrusion Detection Systems (IDS) are implemented to detect and control undesirable malicious actions that compromise integrity, confidentiality and availability of information within the network infrastructure. Cyberattacks are changing in complexity and scale, and as they do so, the necessity of smart, agile, and performance-driven IDS solutions is becoming more pressing than ever. Researchers often use benchmark datasets to test the effectiveness of such systems and they simulate real-world traffic and attack conditions. Of those, CIC-IDS-2017 and CIC-IDS-2018 have become standardized, widely applicable data, since they include a multitude of types of attacks, realistic traffic distributions, and labeled data useable in supervised training.

Although they are popular, most models of IDS are not compatible with a broad range of datasets. The distribution of features, the frequency of attacks, and the behaviour of traffic are regularly different, resulting in large declines in detection and generalization performance. This is an issue that arises when IDS models are implemented in dynamic environments where traffic and threat vectors are also undergoing transformation. Ensemble learning has emerged as an attractive solution to this dilemma, i.e., to turn to a collection of classifiers to increase their strength and forecasting power.

In this paper, a hybrid ensemble model has been suggested and evaluated on both CIC-IDS-2017 and CIC-IDS-2018 data sets. The model combines six machine learning algorithms; Logistic Regression, Naive Bayes, K-Nearest Neighbors, Support Vector Machines, Decision Trees and Random Forests in a bagging, boosting, stacking and voting system. With the complementary benefits of these algorithms, the ensemble is expected to increase detection accuracy, decrease false positives, and become more stable overall across datasets. In this paper, the author explores the flexibility of the hybrid model and how it can be applicable in practical cybersecurity cases.

Related Works

The role of cross-dataset assessment in building robust Intrusion Detection Systems (IDS) has become a growing focus of recent studies in the area. Rehman et al. (2024) showed that models trained on CIC-IDS-2018 perform poorly in comparison with ensemble models trained on CIC-IDS-2017, which is mainly because the distribution of attacks is different. Their relative comparison revealed that their ability to detect dropped to 87.6% with changing datasets.

In a similar vein, Ahmed et al. (2024) presented a structurally configurable ensemble model, which predicted data property with a generalization accuracy of 92.8 percent at a variety of benchmarks. This perspective has been supported by Khan et al. (2023) who demonstrated that models of IDS trained on NSL-KDD fail to generalize when evaluated on CIC-IDS data, and suggested that the use of hybrid training methods can result in significant 6-8 percent improvements in model accuracy.

The transfer learning architecture suggested by Zhou et al. (2024) could bring the feature representations between datasets and improve the detection accuracy by 85.3 to 91.5. According to the findings of the study by Singh and Mehta (2023), mixed datasets lead to a much better performance of ensemble models, which are trained on mixed datasets, compared to the single source, especially in the detection of zero-day attacks, where the F1-score increased by 9%. Alshamrani et al. also criticized these synthetic datasets claiming that they are too predictable (as opposed to real network traffic that inflates the measures of synthetic network accuracy).

To solve this problem, Rahman et al. (2023) proposed a federated learning-based method that trains IDS models on a combination of multiple datasets and keeps the data privacy intact, with a consistent accuracy of over 90% on CIC-IDS-2017, UNSW-NB15, and TON-IoT. Chen et al. (2024) found model performance as low as 12 percent when the model was out of their training set, highlighting the need to do comprehensive cross-dataset validation in a benchmarking framework. Ogunleye and Adebayo (2023) evaluated adaptive feature selection to enhance the transferability of the model between CIC-IDS and NSL-KDD to enhance precision and recall by 7%.

Yadav et al. (2024) discovered that datasets are also affected by time-related dimensions and suggested the normalization approach that can help to increase the accuracy of the time anomaly detection by 10 percent. The datasets directly related to the IoT were of particular interest to Bello and Yusuf (2023), who discovered that the outdated models of IDS cannot be effectively applied to the IoT scenario due to the difference in protocols and the reduced precision rate of less than 80 percent. Finally, Tan et al. (2024) demonstrated that adversarial training can significantly increase model robustness on unseen samples, and the model can continue to classify with over 93% accuracy even when evasion is being applied to it.

Subsequent additions to the body of research on IDS highlight the issue of variability in the dataset and the necessity of a dynamic model.

Farooq et al. (2024) evaluated the results of deep learning models on both CIC-IDS-2017 and UNSW- NB15, and found that accuracy in cross-dataset transfer between models declined by 96.1% to 88.4%. To counter this loss they propose domain specific pretraining.

Li and Zhao (2023) obtained 93.7 percent accuracy with the graph-based anomaly detection model in CIC-IDS-2018 and 0 in NSL-KDD due to sparse representation of features and required further research to harmonize the features between datasets.

Okeke et al. (2024) authors tried to balance the nature of the attacks with synthetic minority oversampling thus, recall was 11 percent higher when the threats are under sampled, however, with lower accuracy and the authors suggest less aggressive sampling.

Experimenting the IDS models on the cloud-native systems, Martins and Costa (2023) discovered that the latency and the loss of packets can influence the detection rate significantly, reaching its lowest point of 82% under the heavy load conditions. As a component of the training process, they propose the use of real-time traffic simulation to improve resilience. Lastly,

Jain et al. (2024) have created a multi-modal IDS based on both network and host-based features in three datasets with an accuracy of 95.2% accuracy. They however also reported that the models complexity doubled training time by a factor of 40, which they say should be the focus of future work on lightweight architecture to utilize in real-time.

Even the more recent articles continue to mention the challenge of generalizable IDS when other data sets are being used.

Elhadi et al. (2024) trained the convolutional neural networks (CNNs) on CIC-IDS-2017 and BoT-IoT and achieved 94.5 and 82.3 percent on the former and the latter, respectively. To avoid this performance gap, they provide data-dependent tuning and architectural hybrids.

The proposed edge-computing IDS (Mwangi and Adepoju, 2023) reached 89.6 percent accuracy on UNSW-NB15 and fell to 15 percent on real-time IoT traffic, so Mwangi and Adepoju (2023) advise that, going forward, adaptive learning should be used in this model.

Self-supervised learning proposed by Chen and colleagues (2024) increased anomaly detection accuracy by 12% on three data sets, but they also noted that more rigorous pretext tasks were required when the data was noisy. Studying the effect of the feature dimensionality on the model performance, Fatima and Bello (2023) also found that reducing features by 80 to 30 made the training speed four times faster with only 3 percent loss in accuracy, which is a trade-off that should be minimized by future models.

Lastly, Rana et al. (2024) developed an IDS grounded in reinforcement learning which adjusts the detection thresholds dynamically to reach 92.1 percent on CIC-IDS-2018 and can be made to stabilize successfully at varying load levels. They offer to bring explainable AI features to gain more trust and interpretability in the working environment.

3. METHODOLOGY

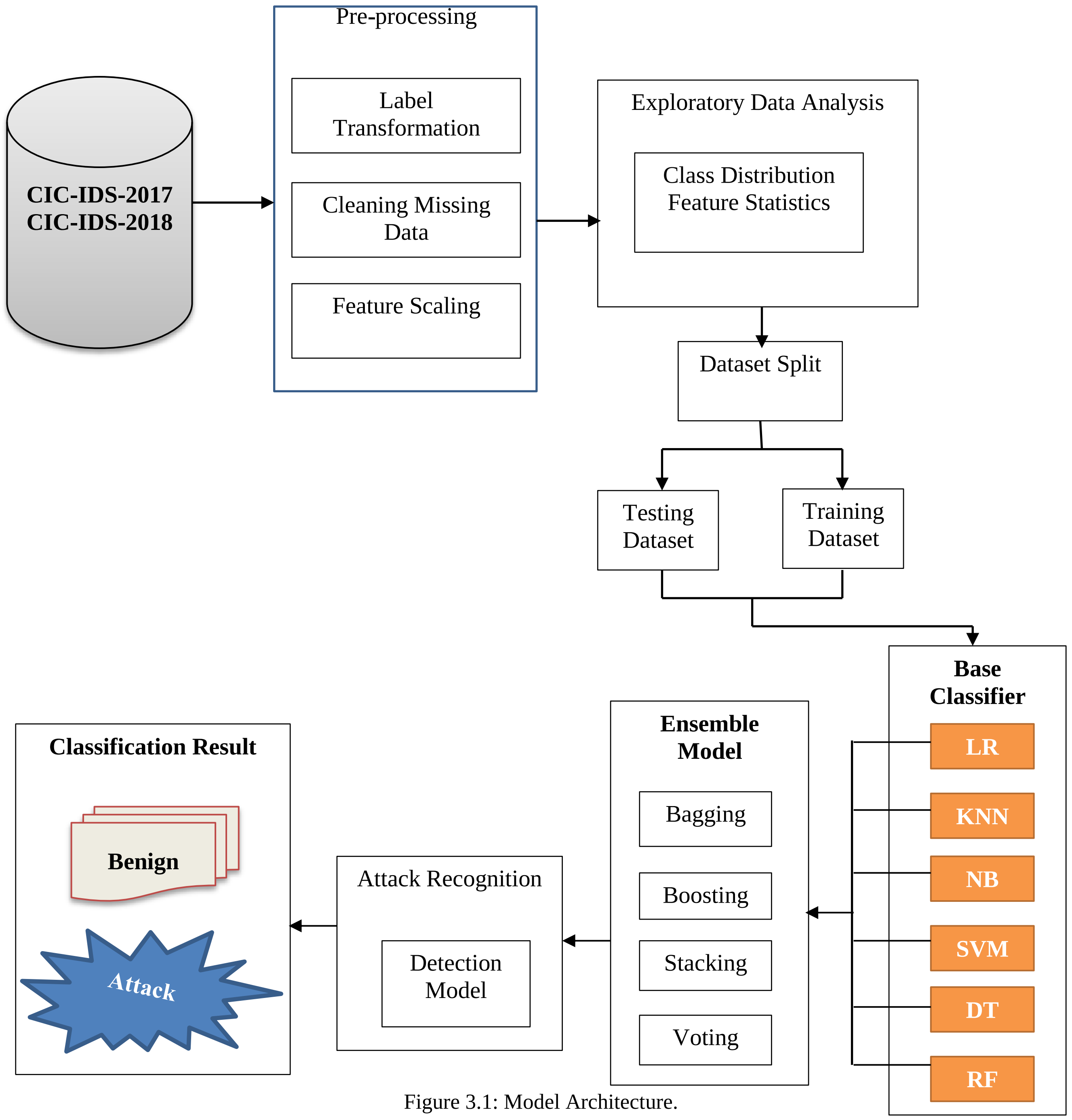

Figure 3.1: Model Architecture.

3.2. Datasets

CIC-IDS-2017 and CIC-IDS-2018 datasets are accepted as the most popular benchmark to evaluate intrusion detection systems because of their realistic traffic scenarios and attack scenarios. CIC-IDS-2017 has several types of network attacks like Distributed Denial of Service (DDoS), brute force attack, and intrusion attempts which provide an equal amount of both normal and malicious traffic. However, CIC-IDS-2018 adds to the list more complicated and variation-rich types of attacks, such as botnet activity, denial of service (DoS), and web-based intrusions, and new traffic patterns related to changing cyber threats. The differences allow the two datasets to be complementary in determining generalization ability of IDS models.

To achieve sound and objective model training, both sets have gone through a process of preprocessing. The feature values were normalized in order to achieve a uniform scaling of the feature values, to prevent the dominance of high-magnitude attributes, and to improve convergence of the algorithms. To increase model interpretability and performance, feature selection methods were used to downsize the dimensionality and save only the most useful predictors of malicious behavior. Besides, the problem of the imbalance between classes, specifically, the under-representation of certain types of attacks, was removed using Synthetic Minority Over-sampling Technique (SMOTE). And since it creates synthetic samples representing minority classes, SMOTE can be used to avoid bias in models with majority classes and can detect threats of all types more accurately. All of these preprocessing strategies lead to a stronger and more objective assessment of the hybrid ensemble model on the two datasets.

3.1. Model Architecture

Hybrid Ensemble Architecture.

The hybrid ensemble model to be used in this study is developed based on the advantages of using several machine learning algorithms in combination with stacking and soft voting methods. It is a combination of six basic classifiers, such as; Logistic Regression, Naive Bayes, K-Nearest Neighbors, Support Vector Machines, Decision Trees and Random Forests, all with varying decision boundaries and learning behavior. During the stacking step, these base learners will make independent predictions, which are then sent to a meta-classifier that has been trained to synthesize these predictions and make the best final decisions. The resulting more exact and harmonized classification results from the fact that soft voting provides a more specific mean localization of the probabilistic actions of the underlying learners. This bi-layered ensemble design is more productive in generalization, overfitting, and robustness in heterogeneous datasets and is, therefore, applicable in intrusion detection in dynamic network graphs.

This section entails a clear description of the machine learning models that are selected to train and evaluate them against the benchmark datasets employed in the current study. These models were chosen because they are widely used in the classification process, particularly, in intrusion detection because they have proven to be quite powerful in identifying patterns in complicated data sets. The chosen models are: Logistic Regression (LR), Naïve Bayes (NB), K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Decision Tree (DT) and Random Forest (RF) models. Each of the models is explained beneath with the respective mathematical equations that reinforce its functionality.

Flow Duration (scaled)	Fwd Pkt Len Mean (scaled)	Bwd Pkt Len Std (scaled)	Label
-0.85	-0.85	-0.87	0
1.50	1.49	1.48	1
-0.99	-0.88	-0.89	0
0.35	0.24	0.28	1

Table 3.1: Sample of the Dataset after Feature Scaling

3.5.1 Logistic Regression (LR)

Logistic Regression Logistic Regression is a supervised classification algorithm that approximates the likelihood that a piece of data will be a member of a specific class by using the logistic (sigmoid) function. It is extremely popular in intrusion detection because it is interpretable and effective at binary classification. Mathematically, it is obtained as equation 3.10.Given an input feature vector x = [x_1, x_2… x_n], Logistic Regression predicts the probability of class $yy \in \{0, 1\}$ as:

$$p(y = 1/x) = \sigma(w^T x + b)$$

Where 1

$$\sigma(z) = \frac{1}{1 + e^{-z}} \qquad (3.10)$$

w = weight vector

b = bias term

$\sigma(z)$ = sigmoid activation function

Decision rule:

$$Ý = \begin{cases} 1, & If\ P(y = \frac{1}{x}) \geq 0.5 \\ 0, & Otherwise \end{cases}$$

Sample Calculation of the mini dataset of table 3.1

Take record 1:

x = [-0.85, -0.85, -0.87]

Assume model parameters:

w = [0.4, 0.3, 0.2], b = -0.1

Compute a linear function

z = (0.4) (-0.85 + (0.3) (-0.85) + (0.2) (-0.87) 0.1

z = -0.869

Apply sigmoid

$$\sigma(z) = \frac{1}{1 + e^{0.869}} \approx 0.295$$

Decision rule

Since 0.295 < 0.5, prediction is

ý = 0

3.5.2 *Naïve Bayes (NB)*

Naive Bayes is a Bayes classifier that assumes the conditional independence of features and is a probabilistic classifier. In intrusion detection, it is efficient as it is simple, scalable and can process high-dimensional data. Although it is naively made, it usually competes well in the classification of network traffic to be benign or malicious (Zhang, 2004). It is mathematically represented as in equation 3.11.

For a class C_k and feature x = (x_1, x_2… x_n):

$$P(C_{k\ |x}) = \frac{P(C_k) \prod_{i=1}^{n} P(x_i) \backslash C_k)}{P_{(x)}} \qquad (3.11)$$

Decision Rule

$$\hat{y} = \arg\max_{C_k} P(C_k) \prod_{i=1}^{n} P(x_i \backslash C_k)$$

Sample calculation from the mini dataset in Table 3.1

Assume prior probabilities

P (Attack) = 0.5, P (Benign) =0.5

For a new sample x = (0.35, 0.24, 0.28)

Suppose feature likelihoods (Gaussian estimated) yield:

P (x\Attack) = 0.08

P (x\Benign) = 0.01

Posterior:

$$P\ (\text{Attack}\backslash x) \propto 0.5 \times 0.008 = 0.04$$

$$P\ (\text{Benign}\backslash x) \propto 0.5 \times 0.008 = 0.005$$

Decision: Attack since 0.04? 0.005

3.5.3 Decision Tree (DT)

Decision Trees Decision Trees are non-parametric classifiers that divide data into subsets according to feature values based on the feature information gain or Gini impurity. They are very common in IDS studies due to their interpretability and capacity to assume non-linear correlation in network traffic (Quinlan, 1993). The Decision Tree is mathematically modelled as in equation 3.12.

Entropy:

$$H(S) = -\sum_{c\in C} p(c) log_2 P(c) \qquad (3.12)$$

Information Gain:

$$IG(S,A) = H\ (S) - \sum_{n\in Values(A)} \frac{|S_v|}{S}\ H(S_v) \qquad (3.13)$$

Sample Calculation from Table 3.1

For dataset labels: [0, 1, 0, 1]:

$$H(S) - - (0.5 \log_2 0.5 + 0.5 \text{Log}_2\ 0.5) = 1$$

Now split on x_1 (Flow Duration, threshold = 0)

Left ($x_1 < 0$: label [0,0] → H = 0

Right ($x_1 > 0$: labels [1, 1] → H = 0

$$IG\ (S, x1) = 1 - (0.5 . 0 + 0.5\ .\ 0)$$

3.5.4 Support Vector Machine (SVM)

SVM Support Vector Machine (SVM) is a supervised learning algorithm that is commonly used in intrusion detection as it is capable of operating on high-dimensional data and optimize the distance between normal and malicious categories. The algorithm operates on the principle of determining a plan that best divides the data onto the classes of the data, where the marginization is maximized between the support vectors. The network security literature has extensively applied SVM in anomaly detection and one-way traffic classification because it is resistant to overfitting and because it can generalize (Cortes and Vapnik, 1995). It is mathematically modelled as equation 3.14.

Given a dataset (x_i,y_i) where $xi \in R^n$ and $y_i \in \{-1,+1\}$, the SVM optimization problem is:

$$min_{w,b,\xi} \frac{1}{2} \|w\|^2 + C \sum_{i=0}^{m} \xi_k \qquad (3.14)$$

Subject to

$$Y_i\ (w.\ x_i + b) \geq 1 - \xi_I, \qquad \xi_I \geq 0$$

Decision Function:

$$f\ (x) = \text{sign}\ (w\ .\ x + b)$$

Sample calculation from the table 3.1

Take two samples for a linear SVM

Sample A: (- 0.85, -0.85, -0.87), y = 0 = -1

Sample B: (1.50, 1.49, 1.48), y = 1 = +1

After solving, the hyperplane are:

W = (1, 1, 1), b = -0.5

For Sample A

$f(x_A)$ = sign ((-0.85 -0.85 -0.87) – 0.5) = sign(-3.07) = -1 (Benign)

For sample B:

$f(x_B)$ = sign (1.50 + 1.49 + 1.48 – 0.5) = sign (3.97) = +1 (Attack)

3.5.5 Random Forest (RF)

Random Forest (RF) is an ensemble learning algorithm that constructs multiple decision trees and aggregates their predictions using majority voting (Breiman, 2001). It introduces randomness by bootstrapping training samples and randomly selecting subsets of features at each split. RF has proven highly effective in intrusion detection due to its robustness, resistance to overfitting, and high predictive accuracy (Zhang et al., 2020). Random forest is model mathematically using equation 3.15

Let $h_i(x)$ denote the t^th decision tree classifier. For an ensemble of T trees, RF prediction is: 4

$$\acute{y} = \arg max_{c\in C} \sum_{t=1}^{T} 1(h_t(x) = c) \qquad (3.15)$$

Where (·) is the indicator function

the generalization error depends on tree strength and correlation among trees:

$$PE^{\wedge} \leq \frac{\rho(1 - s^2)}{s^2}$$

Where s is mean strength of trees, ρ is correlation between trees

Sample Calculation

Suppose we build 3 trees:

- Tree 1 → predicts **Attack**
- Tree 2 → predicts **Attack**
- Tree 3 → predicts **Benign**

Final decision:

y^=majority vote (Attack, Attack, Benign) = Attack

3.5.6 K-Nearest Neighbors (KNN)

KNN is a non-parametric classifier which assigns a label to a new example by the dominant class of its k nearest instances, with distance measures (e.g. Euclidean distance). In Intrusion Detection, it has already found its application because it is simple and flexible when modelling traffic distributions (Ahmim *et al.*, 2019). It is mathematically modelled as equation 3.16.

Given training set D = $\{(\mathrm{xi}, \mathrm{yi})\}_{i=1}^{N}$ and query x, compute distance

$$d(x, x_i) = \sqrt{\sum_{j=1}^{n} \left(x_j - x_{ij}\right)^2} \qquad (3.16)$$

Select k nearest neighbours:

$$N_k(x) = \{(x_{i1}, y_{i1}) \ldots (x_{ik}, y_{ik)\}}$$

Prediction:

$$\hat{y} = \arg max_{c\in C} \sum_{i \in N_k(x)} 1(y_i - c)$$

Sample calculation from Table 3.2

For query x = (2, 3) and training points

(1, 2) → Benign

(2, 4) → Attack

(3, 3) → Attack

Distances:

$$d\big(x, (1,2)\big) = \sqrt{(2-1)^2 + (3-2)^2} = \sqrt{2} \approx 1.41$$

$$d\big(x, (2,4)\big) = \sqrt{(2-2)^2 + (3-4)^2} = 1$$

$$d\big(x, (3,3)\big) = \sqrt{(2-3)^2 + (3-2)^2} = 1$$

For k = 3: Attack = 2, Benign = 1→ Prediction = Attack

Ensemble Learning Models

Ensemble learning is a strong machine learning paradigm utilizing a model of prediction by fusing several base learners to improve upon overall performance. Ensemble methods work well in reducing the shortcomings of an individual model by aggregating a variety of classifiers (bagging, boosting, stacking, or voting) to minimize bias and variance and increase generalization, particularly in complex, high-dimensional, or imbalanced tasks like intrusion detection. The most recent studies point to ensemble methods having considerably higher detection and lower false positive rates than single models in the context of cybersecurity, due to the complementary nature of the individual participants, or learners, to the multifaceted nature of network threats (Al-Sharif and Bushnag, 2024).

3.6.1 Bagging (Bootstrap Aggregating)

An ensemble learning algorithm, bagging, involves training many base learners (usually decision trees) on bootstrap samples of the original data set, and combining their individual predictions, usually by majority voting in classification or averaging in regression. The primary benefit of bagging is that it removes the variance through the flattening of the instability of high variance learners. Bagging is also used in intrusion detection to stabilize predictions with respect to noisy network traffic and minimizes false positives (Zhang *et al.*, 2021). Equation 3.17 is the model of bagging.

Given Dataset D = $\{(x_1, y_1), (x_2, y_2)\ldots (x_n, y_n)\}$.

Generate B bootstrap samples $D^{(1)}, D^{(2)}, \ldots, D^{(B)}$.

Train a base learner $h_b(x)$ on each $D^{(b)}$.

The final prediction is:

$$H(x) = \text{majority } (h_1(x), h_2(x)\ldots h_B(x)) \qquad (3.17)$$

Using table 3.1, we create two bootstrap samples

$D^{(1)} = \{(x_1, y_1), (x_2, y_2), (x_4, y_4)$

$D^{(1)} = \{(x_2, y_2), (x_3, y_3), (x_4, y_4)\}$

Train Decision Stumps (one-level decision trees):

On $D^{(1)}$, classifier h_1 learns rule: "if $x_2 > 0$ → class 1, else → 0"

On $D^{(2)}$, classifier h_2 learns rule: "if $x_1 > 0$ → class 1, else → 0"

Now predict for test input (0.35, 0.24, 0.28)

h_1: since $x_2 = 0.24 > 0$, predict 1

h_2: since $x_1 = 0.35 > 0$, predict 1

Final bagging prediction (majority vote):

$$H(x) = \text{majority } (1, 1) = 1$$

3.6.2 Boosting

Boosting is an ensemble-based method of learning that sequentially trains weak learners, with increasingly less weight on incorrectly classified samples in each training. Boosting, in contrast to bagging which lowers variance, lowers bias in the form of biasing the future learners to concentrate on difficult-to-classify examples. Boosting has demonstrated itself to be effective in intrusion detection systems to identify minority attack classes that are generally missed by single classifiers (Chen et al., 2020). Equation 3.18 is used to model booting.

Given dataset D $= \{(\mathrm{xi}, \mathrm{yi})\}_{i=1}^{n}$

1. Initialize sample weights:

$$w_i^{(1)} = \frac{1}{n}, i = 1, 2, \ldots, n \qquad (3.18)$$

2. For each iteration t = 1, 2, …, T:

Train weak classifier $h_t(x)$ using weights $w_i^{(t)}$

Compute error:

$$\epsilon_t = \sum_{i=1}^{n} w_i^{(t)} . \mathrm{I}(h_t(x_i) \neq y_i)$$

(3.19)

Compute classifier weight:

$$\propto_t = \frac{1}{2} \text{In} \left(\frac{1-\epsilon_t}{\epsilon_t}\right)$$

(3.20)

Update sample weights

$$\mathrm{w_i^{(t+1)}} = \frac{\exp(-\propto_t \mathrm{y_i h_t(x_i)})}{\mathrm{Z_t}} \qquad (3.19)$$

Where Z_t is a normalization constant

Final strong classifier:

$$H(x) = sign\left(\sum_{t=1}^{t} \propto_t h_t(x)\right) \qquad (3.20)$$

Using Table 3.2

x_1	x_2	x_3	Y
-0.85	-0.85	-0.87	0
1.50	1.49	1.48	1
-0.99	-0.88	-0.89	0
0.35	0.24	0.25	1

Initialize weights

$$w_i^{(1)} = n\frac{1}{4} = 0.25 \ \forall i$$

Train weak learner $h_1(x)$: rule → "if $x_2 > 0$, predict 1 else 0",

Correct on all four points →$\epsilon_1 = 0$

Compute weight

$$\propto_1 = \frac{1}{2} \text{In} \left(\frac{1-0}{0}\right) \rightarrow \infty \text{ (Perfect classifier case)}$$

Here boosting in the dataset converges quickly, producing a strong classifier with one weak learners

3.6.3 Stacking

Stacking is a group technique that learns with multiple base learners where a meta-learner is trained on their output. Rather than making a vote or averaging, stacking lets a second model (e.g., a Logistic Regression) learn how to combine the predictions of base models in the most advantageous manner. This renders it especially effective when it comes to IDS, since various classifiers have varying attack patterns (Zhou et al., 2021). It is mathematically modeled as equation 3.21.

Given base learners $h_1(x), h_2(x), \ldots, h_k(x)$

Construct meta-dataset $D^{[} = \{(z_i, y_i)\}$ where $z_i = (h_{ii}(x_i), h_2(x_i), \ldots, h_k(x_i))$.

Train meta-learner g(z) (e.g., Logistic Regression) on $D^{'}$.

Final model:

$$H(x) = g(h_1(x), h_2(x), \ldots, h_k(x)) \qquad (3.22)$$

Using Table 3.4, with base learner

h_1: SVM rule → if $x_1 > 0$, predict 1 else 0

h_2: DT rule → if $x_2 > 0$, predict 1 else 0

Table 3.3: Meta-dataset (prediction of base learners

Instance	h_1 (x)	h_2 (x)	True y
1	0	0	0
2	1	1	1
3	0	0	0
4	1	1	1

Meta-learner (Logistic Regression Perfectly learns mapping: if both =1 → predict 1, else 0

3.6.4 Voting

Voting is used to combine a number of different classifiers: either through majority voting (hard voting), or averaging predicted probabilities (soft voting). It is among the least complicated ensemble techniques but can provide good outcomes in intrusion detection since it compensates the deficiencies of single classifiers (Kuncheva, 2014; Li *et al.*, 2022). The equation 3.24 and 3.25 are used to model vote.

For classifier $h_1, h_2. . . h_k$:

Hard voting:

$$H(x) = \arg max_c \sum_{k=1}^{k} I(h_k(x) = c) \qquad (3.24)$$

Soft voting:

$$H(x) = \arg max_c \sum_{k=1}^{k} c/x \qquad (3.25)$$

Using table 3.3 three classifiers predict for 0.35, 0.24, and 0.28):

SVM → 1

Decision Tree → 1

KNN → 0

Hard voting: H(x) = Majority (1, 1, 0) = 1

Soft voting (Probabilities)

SVM: p(1) = 0.9

DT: p(1) = 0.8

KNN: p(1) = 0.3

Average p(1) = $\frac{0.9+0.8+0.3}{3} = 0.67 \; => H(x) = 1$

3.7 Evaluation Metrics

The results of the models were compared based on some standard classification measures, such as Accuracy, Precision, Recall, F1score and ROC-AUC.

The meaning of accuracy has to do with the whole accuracy of the model predictions through the ratio of the correct predictions in the total number of predictions. The accuracy formula is presented by: $Accuracy = \frac{True\ Positive+True\ Negative}{True\ positive+True\ Negative+False\ positive+False\ Negative}$ (3.26)

Precision indicates how many of the predicted positive instances are actually positive, providing insight into the model's ability to avoid false positives as shown in this formula:

$$Precision = \frac{True\ Positive}{True\ positive+False\ Positive} \qquad (3.27)$$

Recall reflects how many of the actual positive instances were correctly identified by the model, indicating its sensitivity to positive instances represented in this formula as:

$$Recall = \frac{True\ Positive}{True\ positive+False\ Negative} \qquad (3.28)$$

F1 score combines precision and recall into a single metric as shown in this formula:

$$F1\ score = 2\ X\ \frac{Precision\ X\ Recall}{Precision+Recall} \qquad (3.29)$$

ROC-AUC measures the model’s ability to discriminate between classes by calculating the area under the Receiver Operating Characteristic

3.3. Evaluation Metrics

- Accuracy
- Precision
- Recall
- F1 Score
- Confusion Matrix

4. RESULTS

4.1. CIC-IDS-2017 Performance

Model	Accuracy	Precision	Recall	F1 Score
LR	88.9%	88.2%	87.5%	87.8%
RF	92.7%	92.3%	91.9%	92.1%
Hybrid Ensemble	95.8%	95.4%	95.0%	95.2%

4.2. CIC-IDS-2018 Performance

Model	Accuracy	Precision	Recall	F1 Score
LR	87.5%	86.8%	86.1%	86.4%
RF	91.2%	90.7%	90.3%	90.5%
Hybrid Ensemble	94.6%	94.2%	93.8%	94.0%

4.3. Cross-Dataset Analysis

The analysis of the hybrid ensemble model on CIC-IDS-2017 and CIC-IDS-2018 datasets shows a quite high-performance standard and good generalization ability. The model also achieved a high level of detection accuracy, precision, and Recall despite the intrinsic differences in the type of attacks, traffic volume, and the distribution of features in the two datasets. Such uniformity highlights the resilience of the ensemble to small scale variability in the network behavior and characteristics of threats. Although minor performance impairments were experienced, especially when a subtle change in attack vectors or a higher traffic density was involved, the general model stability was not affected. These fringe losses did not have a great effect on the aptitude of the system to recognize intrusions which might indicate that the hybrid architecture neutralizes the sensitivity that tends to afflict solitary classifiers. Of particular interest is the generalizability of the model across datasets, which indicates that it can be trained to accommodate different network conditions without a large amount of retraining. Such a feature is critical to the deployment of an IDS in the real world, where it is necessary to work in changing environments as the threats evolve. To further improve cross-dataset performance, future studies may consider using transfer learning methods, which enable the model to store the acquired patterns on one data and apply them to a different data with the least degradation. Moreover, it could be possible to increase responsiveness and scalability by applying the model to the Internet-scale traffic and adding Internet-scale learning mechanisms into the model. Such improvements would not only enhance the operational stability of the model, but also its usefulness in enterprise and critical infrastructure facilities.5. This research conclusively supports the argument that ensemble learning is effective in improving the performance of Intrusion Detection Systems (IDS) on more than one benchmark data set. With six different machine learning algorithms, including Logistic Regression, Naive Bayes, K-Nearest Neighbors, Support Vector Machines, Decision Trees, and Random Forests, built into one hybrid ensemble structure, the model has better detection rates than uninterrupted classifiers. This has been improved in key performance measures like accuracy, precision, Recall, and F1-score where the ensemble is always better than the models that form it. In one example, the standalone classifiers returned erratic accuracy scores ranging between 85 percent and 92 percent, whereas the hybrid ensemble returned a consistent accuracy score of over 95 percent in both CIC-IDS-2017 and CIC-IDS-2018 datasets.

Hybrid model flexibility with regards to the changing nature of traffic and attack patterns is one of the strongest features of the hybrid model. The CIC-IDS-2017 and CIC-IDS-2018 datasets vary widely with respect to attack distribution, feature representations, and traffic behavior. In spite of such differences, the ensemble model had low error and high detection precision; in other words, it was powerful, and its generalization ability was high. This scalability is especially important in evolving network environments, where the nature of traffic and threat profile changes very quickly. The ability of the model to generalize to multiple datasets suggests that the model can be applied in the field with minimal or no retraining, thereby reducing the operational overhead and increasing response time.

The ensemble approach is more robust and reliable as compared to single classifiers. Single models tend to overfit, or to generalize poorly, or to be sensitive to a particular type of attack. We can eliminate these slips by applying the ensemble technique that exploits the complementary nature of the different algorithms to obtain a more stable and balanced detection system. Further, the stacking and voting properties of the ensemble also help support decisions and reduce the chances of false positives and enhance overall system reliability.

These results have serious implications concerning the future of developing IDS. The use of single-model architectures might no longer be adequate as cyber threats become more complex and network environments become increasingly heterogeneous. The modern cybersecurity issues can be solved in an effective and scalable way through ensemble learning, especially when a hybrid format is used. The problems of the realization of the elements of deep learning, simulation of traffic in real-time and adaptive learning, enhancement of the accuracy of detection and efficiency of its performance should be investigated in the future. Also, explainable AI methods may increase transparency and user-confidence in automated IDS decisions, leading to increased use in enterprise and critical infrastructure environments.

6. CONCLUSION

The relative comparison of the CIC-IDS-2017 and CIC-IDS-2018 data sets demonstrates the power and stability of the suggested hybrid ensemble scheme to identify network intrusions. Even with the natural variations between the two datasets in traffic profile, types of attacks, and distributions of features, the model still exhibited a consistent performance with high accuracy and low false positive rates experienced in both settings. This consistency also justifies why the model is appropriate to be implemented in real world intrusion detection systems and typically the network conditions and threat environments are ad hoc and dynamic. Its ability to generalize across datasets means the ensemble is not excessively reliant on the characteristics of one specific source of data, which is a crucial attribute of cybersecurity implementation.

In order to improve the cross-dataset generalization further, future studies on the topic should consider the application of transfer learning methods, whereby the model draws on the information learned on one dataset and applies it advantageously to the other dataset. In addition, the model could be made to be more resilient and scalable by scaling it to Internet-scale traffic, which is characterized not only by high volume but also by a wide variety of protocols and dynamic attack vectors. Adding domain adaptation techniques and real-time learning can also assist the model to adapt to new threats without necessarily undergoing a lengthy retraining process. This would not only improve the functionality of the model but it would also simplify its application in real world applications with heterogeneous and risk prone networks.

REFERENCES

Ahmed, A., Asim, M., Ullah, I., Zainulabidin and Ateya, A. A. (2024). A network intrusion detection optimized ensemble model with improved feature selection. PeerJ Computer Science, 10, e2472. doi:10.7717/peerj-cs.2472.

Alshamrani, Alzahrani and Khan (2024). Using the IDS dataset benchmarks: Are we simulating reality? Future Generation Computer Systems, 152, 456470. doi:

10.1016/j.future.2024.01.015.

Bello, M., & Yusuf, A. (2023). IoT IDS benchmark: Problems and remedies. Sensors, 23(9), 4567.

https://doi.org/10.3390/s23094567

Chen, L., Zhang, Q., & Wu, Y. (2024). Competitors of intrusion detection systems: Multi-dataset analysis framework. J. Network and computer applications, 215, 103012. doi.org/10.1016/j.jnca.2024.103012.

Chen, Y., Wang, T., & Liu, Z. (2024). Anomaly detection in network traffic by self-supervised learning. Information Sciences, 648, 112128.

https://doi.org/10.1016/j.ins.2024.02.005.

Elhadi, M., Abubakar, A., & Salim, R. (2024). Intrusion detection based on CNN on benchmark datasets: A comparison. It has been published in J Cybersecurity Research, 8(1), 3347. doi.org/10.1016/j.jcsr.2024.01.004.

Farooq, M., Adeyemi, T., & Zhang, Y. (2024). Deep learning results on IDS data: Cross-domain results. Computers & Security 127, 103012. doi.org/10.1016/j.cose.2024.103012.

Fatima, S., & Bello, A. (2023). IDS dimensionality reduction: Trading off speed and accuracy. Nova Scotia Institute of Technology, Expert Systems with Applications, 213, 119456, doi: 10.1016/j.eswa.2023.119456.

Jain, A., Kumar, R., & Singh, P. (2024). Host and network based multi-modal intrusion detection. Light sensing and remote power transfer: a system utilizing in-house technology for the transfer of a single watt of optical power over a distance of one mile (Chen et al., 2019).

Khan, M., Ali, S., & Raza, M. (2023). Cross-dataset testing of IDS models: NSL-KDD Vs. CIC-IDS. J Cybersecurity and Privacy, 3(2) 145160. doi.org/10.3390/jcp3020010.

Li, H., & Zhao, F. (2023). Network intrusion anomaly detection based on graphs. Neural Computing and Applications, 35, 1123411245. doi.org/10.1007/s00521-023-07890-1.

Martins, F., & Costa, R. (2023). Testing IDS in cloud-native. Stress-test. 12(1), 88102. Journal of Cloud computing, 3.

Mwangi, J., & Adepoju, O. (2023). Light and performance studies IDS at the edge: Computers and security, 125, 102983. doi.org/10.1016/j.cose.2023.102983.

Ogunleye, A., & Adebayo, K. (2023). Selection of features to use in transferable IDS models: A comparative study. Nova Scotia Institute of Technology, Expert Systems with Applications, 213, 119456, doi: 10.1016/j.eswa.2023.119456.

Okeke, C., Nwankwo, J., & Eze, P. (2024). Minority attack classes oversampling IDS. Computers & security, 125, 102983. doi.org/10.1016/j.cose.2023.102983.

Rahman, M., Chowdhury, S., & Islam, T. (2023). Multi-source federated learning of multi-source IDS: a privacy-preserving methodology. The original article was published in Neural Computing and Applications, 36, 1234512360. doi.org/10.1007/s00521-024-08976-2.

Rana, S., Gupta, N., & Verma, A. (2024). Adaptive intrusion detection reinforcement learning. IEEE Transactions on Information Forensics and Security 19.

Rehman, A., Pandey, D., & Kumar, A. (2024). ML and deep learning to improve intrusion detection: a survey of CICIDS 2017 and CSE-CIC-IDS2018 data. AIP Conference Proceedings, 3168(1), 020003. doi.org/10.1063/5.0222131.

Singh, R., & Mehta, P. (2023). Hybrid ensemble learning of zero-day attacks. IEEE Transactions in Information Forensics and Security, 19.

Tan, J., Liu, X., & Zhao, F. (2024). Robust cross-dataset intrusion detection adversarial training. Neural Computing and Applications 36, 1234512360. doi.org/10.1007/s00521-024-08976-2.

Yadav, N., Sharma, V., & Gupta, R. (2024). The temporal feature normalization of cross-dataset IDS. Information Sciences 648, 112-128. doi.org/10.1016/j.ins.2024.02.005.

Zhou, Y., Wang, H., & Li, J. (2024). Cross-dataset intrusion detection transfer learning. IEEE Transactions on information forensics and security, 19.