International Journal of Engineering Trends and Technology
ISSN: 2231-5381/ https://doi.org/10.14445/22315381/IJETT-V74I11P133

Volume 74 Issue 11, 521-537, November 2026
© 2026 Seventh Sense Research Group®

Review Article

Optimised Intrusion Detection Framework: Feature Selection and Hyperparameter Tuning for Enhanced Machine Learning Performance

Sayyada Mubeen[1], Balakrishna Gudla[2]

[1,2]*Department of Computer Science and Engineering, Malla Reddy University, Hyderabad, Telangana, India.*

[1]*Corresponding Author: 2232cs010018@mallareddyuniversity.ac.in*

Received: 21 May 2026 Revised: 31 August 2026 Accepted: 12 September 2026 Published: 25 November 2026

Abstract *- The increasing number of cyberattacks has led to a never-ending evolution of improving cybersecurity. Heuristics designed to detect intrusions have been the standard for many traditional security systems. Still, with Machine Learning (ML) becoming more prominent, learning-based models can perform out of the box by scaling with data available through continuous training on labelled data. However, the downside of supervised ML is that it does not perform well in intrusion detection if the training data is either insufficient or of poor quality. To tackle these difficulties, it is essential to improve ML models. In this task, we investigate the establishment of an IDS that incorporates feature engineering techniques and is based on machine learning. This paper proposes two algorithms: Learning-based Intrusion Detection (LbID) and Hybrid Feature Selection (HFS). Using the CICIDS2017 dataset, the system was tested for Classification challenges that are binary and multi-class. The RF model attained the maximum accuracy (94.22%) in binary classification. The Decision Tree model performed the best without optimisations and delivered 91.67% accuracy in multi-class classification. Using the tested optimisation techniques, RF achieved the best accuracy at 93.46%. Additionally, these findings contribute to our knowledge of the fusion of feature engineering and ML in improving the performance of ID.*

Keywords *- IDS, ML, Feature selection, Hyperparameter tuning, CICIDS2017 dataset.*

1. Introduction

The transformation of cyberspace in the modern era makes maintaining cybersecurity a daunting task (Razan et al. 2019). Traditional security cannot protect against advanced attacks, so ML is essential Ahmad et al. (2021). Using ML, an IDS can gradually identify known and unknown assaults. Research on improving cybersecurity is crucial due to the rise in cyberattacks.

Improving the intrusion detection system is not a one-person show; maintaining security is a constant Endeavour. This is the reason behind this work, which is significant for what it aims to indicate on appropriate models for ID, optimisation processes, and training data. This project aims to offer a computational framework for effective intrusion detection through machine learning and optimisations. Here are some questions for study.

Q1: Can intrusion detection systems be developed using ML models? RQ1 and RQ2 confirm the importance of feature selection and hyperparameter tuning for ML models' ability to detect intrusions.

The results drawn from the literature form the basis of the research questions. The first question examines the effectiveness of ML models in ID, and the second discusses two modifications to ML models to boost intrusion detection effectiveness. This study uses an empirical research methodology. These are the primary contributions made by this work. First, it creates and uses an ML-based ID framework. Second, A variety of ML models are evaluated in terms of ID, both for binary (binomial) and multi-class classification, with hyperparameter tuning and best-practice feature engineering (e.g., scaling, encoding, dimensionality reduction). Two algorithms are presented and identified: LbID and HFS. The document's remaining sections are organised as follows. Section 2 examines the research on many ID techniques currently in use. This section describes the proposed approach to investigating the phenomenon in Section 3. Section 4 discusses the design specification. Sections 5 and 6 provide information about the implementation and evaluation. Section 6 offers an additional assessment of the relevance of this research. This section finishes with the findings and recommendations for future research.

This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/)

2. Related Work

Lee and others. On the other hand, ML improves network ID systems' accuracy without requiring specific knowledge [1]. The C-ELM approach proposed fast learning and high attack detection rates by adding hidden neurons step-wise.

Abdulhammed et al. Threat-preferring [2] Network Security is important as threats are increasing. Machine learning also helps IDS to reduce features with the help of Auto-Encoder and PCA, thereby increasing DR, F-Measure, FAR, and accuracy. Alhajjar et al. Metrics that the research encourages for performance comparison include Network Intrusion Detection Systems (NIDS) threats and the use of DL and evolutionary computing techniques in combination with CombinedMc (which has a greater accuracy rate on CICIDS 2017) [3].

Results indicate that NIDS are vulnerable to hostile disruption due to high levels of misclassification. Bertoli et al. [4] addressed the problem of network ID with several years of outdated datasets. The new 5-phase model deployment procedure provides the best model performance with optimised resource usage, more categories, improved dataset production, and testing of lightweight operating systems among the planned initiatives. Ahmad et al. [5, 6] examined both advantages and disadvantages of ML and DL-based NIDS techniques. It addresses present advancements, outlines dataset limitations, and proposes future research directions for more compact and performant DL-based NIDS.

Zaman et al. This work relied on detecting abnormal behaviour in network traffic [6]. Due to the limitations of signature-based IDS, ML-based methods have emerged. Using the Kyoto 2006+ dataset, RBF outperforms the other seven machine learning techniques analysed in this research.

An ensemble technique for further study also yields good results. Sultana et al. For improved protection, ML and DL approaches are partitioned to SDN-based NIDS [7]. This study examines the most recent ML/DL work in SDN-based NIDS, emphasizing the problems and possible solutions.

Li et al. Facebook antagonistic literature review [8]: Predicting nodes and links is an important part of network attack defence, and ML and data mining techniques are used in network attacks. Research is done using the KDD 99 dataset and evaluates fuzzy logic and ANN. Regarding AI-based defence against cyberattacks, the article outlines the problems and future research areas. An ID system based on ML stacking ensembles is proposed in this article in response to the requirement for one by Parashar et al. [9]. Random Forest, XGBoost, and ID3 all did well. Taher and others. In this paper [10], ANN with wrapper feature selection and the supervised machine learning system outperformed SVM in identifying whether network traffic was benign or harmful.

Carrion and associates. This work aims to improve NIDS assessment through a systematic technique utilising the UGR'16 dataset. Network Intrusion Detection Systems (NIDSs) are essential in today's networked environment, yet there are no recognised methods for evaluating them. Halimaa and collaborators. Because they monitor and remove anomalies, IDS are crucial to network security [12]. ML techniques such as SVM and NB increase the IDS's detection precision. Phadke et al. [11]. However, the increased threats from the internet have given rise to the need for dynamic intrusion detection systems based on ML [13].

Various methods enhance the accuracy of Network Intrusion detection. Dini et al. KNN outperforms ANN in this dataset by a narrow margin [14], where feature selection is essential to achieve accurate ID. ML is increasingly used to detect anomalous behaviour, including network infiltration. KNN and ANN approaches are employed to protect LAN traffic analysis. Mishra and others. These techniques are examined for each type of attack, focusing on specific strategies to assess limits for the particular kind of algorithm using machine learning to identify intrusions within [15].

Costa et al. The other is paranoid speculations about security on a global level, upon which [16] invested in state-of-the-art IDS. It concentrates on accuracy and efficiency gains while rejecting the problems of IoT machine learning. Seraphim et al. Awareness of IDS has been sparked due to technological advances [17]. Numerous research studies have used ML to identify network security flaws. Several techniques are investigated, including k-means, SVM, and Random Forest. This two-level approach uses basic and advanced learning algorithms to enhance IDS performance as top and bottom layers, including Artificial Neural Networks (ANN). Devi & Co. Age-related digital expansion [18] emphasises the necessity of automated security. Intrusion Detection Systems (IDS) are primarily responsible for identifying threats, whereas Network Intrusion Detection Systems (NIDS) offer network defence. For example, the NSL-KDD Cup 99, KDD 99, and DARPA 1999 datasets are out of date and will not be used to evaluate newly available attacks for ten years. This study analyses the CIDDS-001 dataset, learns new techniques, and chooses the Verma et al. (aligned decoder) model. Almseidin et al. is the most recent favourite application of their machine learning technique [19]. IDS counterattacks; however, there is an issue with tactical shifting. KDD dataset: Most related research focuses on FP and FN. The decision table and RF performed well in terms of attack distribution. The importance of precise intrusion detection was shown in tests with 60,000 data points. Liu et al. This leads to their hiding due to non-uniform network load, so they need to be detected by NIDS [20]. The DSSTE algorithm corrects imbalance by creating new samples and improving the classification. Evaluations of the NSL-KDD and CSE-CIC-IDS2018 datasets using DSSTE outperform state-of-the-art ones, as shown by SVM and RF models.

Wu and others. Using machine learning, Zhang et al. (2022) developed a network IDS that is both accurate and efficient [30]. To reduce false alarms, it uses a random forest. Intrusion Detection Systems (IDS) leverage cyber threats such as anomaly and signature detection to identify intrusions, even if Megantara and Ahmad [22] made improvements in many industries possible. In this paper, a mixed ML approach is proposed, including feature selection and feature reduction, which enhances the precision of R2L attack detection. [23] Reducing outliers and unbalanced data still poses a problem in improving IDS and needs more research. Software-Defined Networks (SDN) by Abubakar and Pranggono [24] also generate security problems.

So, IDS must be integrated with SDN. Machine learning and flow-based IDS could enhance the IDS testbed Snort utilises to identify threats. Khan and Gumaei [25] evaluated artificial intelligence classifiers for network ID in a different study, concentrating on accuracy and efficiency. On the KDD99 and UNSW-NB15 datasets, KNN, DT, RF, and HT all perform well in 10-fold cross-validation with test models.

NID-Shield (Rincy and Gupta [26]) is a predictive model that employs ML methods and CPPER for feature selection, providing a unified intrusion detection system. Low FPR and encouraging accuracy are achieved through evaluations on the NSL-KDD and UNSW-NB15 datasets.

Li et al. [27] enhanced machine learning: Feature extraction > Feature reduction. Sensitivity: Less Sensitivity; Detection. This enables faster training and larger improvement margins. Jaradat et al. Machine learning is a process in which a model searches the network for irregularities; [28] is an example. They used classification. Intrusions were detected using feature selection and three classifiers.

Findings determine the accuracy rate. The top-performing algorithms include XGBoost, Random Forest, Decision Tree, and Network monitoring (Fan and You [29]). Prevent overfitting by using integrated models. CatBoost and Logistic Regression are Feature-Sensitive and effective. Support vector machines and plain Bayesian models are crappy. Literature on Security Advantages derived from results. Zhang et al. Random forest and decision trees were combined to obtain optimal precision and recall within the scope of network intrusion detection problems [30].

Talukder et al. Cybersecurity also requires network ID with ML-based analysis [31]. The model leverages PCA, RO, and clustering and is better in terms of accuracy. Enhances security posture and reduces false alert rates. Dhaliwal et al. On the NSL-KDD dataset, XGBoost minimises the impact of network interface data to enhance data integrity [32]. Network security is an increasingly crucial aspect of technology as it advances. Zhang et al. The IDTSRF model improves precision, recall, and accuracy with decision boundary entropy [30].

The volume of data and the significance of features make network intrusion detection difficult. Using feature selection and imbalance handling, Chimphlee [33] introduced a two-step process to identify irregularities in network data. Gao et al. [38] suggested an Adaptive Ensemble ML model that includes DT, RF, KNN, DNN, and MultiTree with Adaptive Voting and tested the model with the NSL-KDD dataset. Their technique got an accuracy of 85.2% and proved that the use of ensemble learning in ID systems is beneficial.

However, the relatively low accuracy shows potential for further feature and model optimisations. Wei et al. [39] presented the implementation of an optimised Deep Belief Network (DBN) based on two Emperpkan learning algorithms - AFSA and GA, along with PSO, for the purpose of enhancing intrusion classification. With five categories, they got an accuracy of 82.36% using their approach on the KDDTest+ dataset.

However, the attacks from minority classes like U2R and R2L were still a problem. In [40], Ali et al. implemented a FLN optimised with PSO for ID in the KDD Cup 99 dataset. They proved their results and showed that PSO is effective in improving FLN-based Intrusion Detection (ID), though they used the relatively old KDD Cup 99 benchmark. Shetty et al. [41] suggested an explainable Random Forest-based IDS on the CICIDS2017 dataset with 92% accuracy in classifying benign and attack traffic. The study showed that SHAP can be employed to gain insights into the role network-flow features play in the prediction of intrusions.

3. Proposed Framework

This research aims to achieve intrusion detection by optimising the machine learning framework. The research methodology employed to accomplish the study's goal is depicted in Figure 1.

Step 1: Locate and Examine Datasets. This study was conducted by analysing various datasets; CICIDS2017 is relatively new, offers a variety of intrusion types, and provides excellent benchmarking capabilities. CICIDS2017 is, therefore, the empirical study dataset. XGBoost, RF, DT, and several ML models, including Extra Trees, were judged appropriate for this study.

All of these supervised learning techniques rely on trees. These models work well when trained on good-quality data. Otherwise, they become less and less effective. The literature review suggests that the problem is addressed by examining feature selection and hyperparameter tuning to utilise ML models effectively. Figure 1 presents the design of the proposed IDS, based on the methodology results.

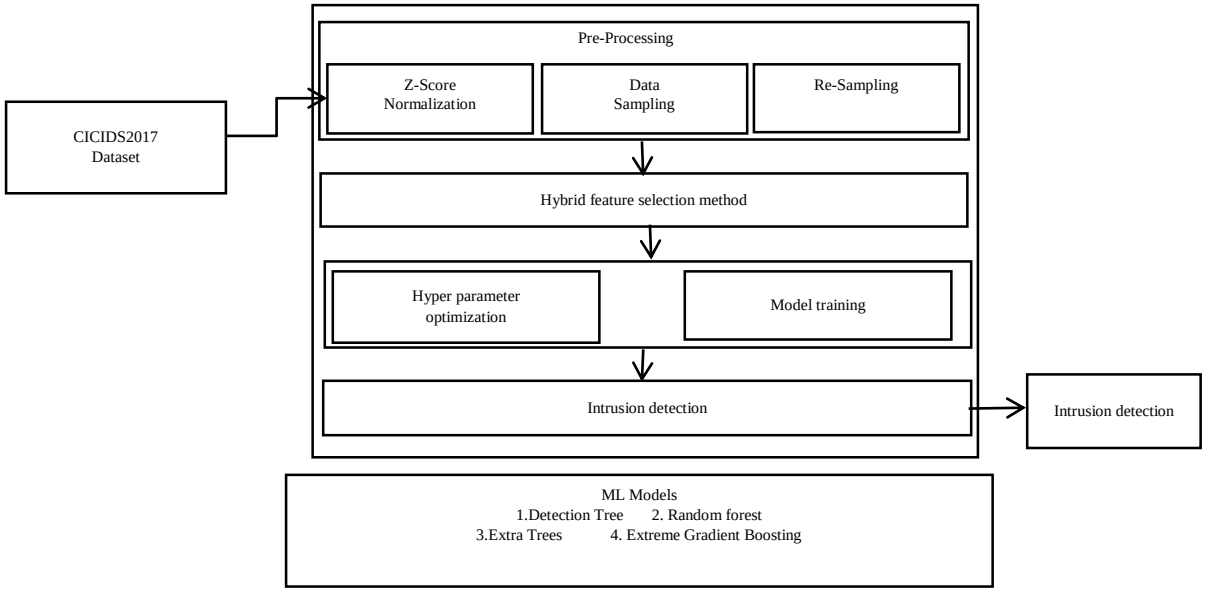

Fig. 1 Architecture of the proposed intrusion detection system

Here is how the proposed system works. The data is pre-processed so that the provided dataset is ready for use; The train–test split is performed after the train–test split and only T_1 is resampled with SMOTE in order to solve the class imbalance issue; the resampled data are zeroed out prior to Z-score-based normalisation; T_2 is not resampled and is not used for SMOTE; zeroing out blank values does not cause a problem because these are not used for SMOTE.

After the preprocessing phase, the data is subjected to feature engineering in order to determine which features perform the best. The hyperparameters corresponding to each machine learning model utilised in the empirical study are then refined.

Hyperparameter tuning is a process that improves the performance of a particular ML model by determining the best values for its hyperparameters. The next step is training the models with the TD. When training is complete, the learned model is released for use later. The trained model is evaluated against the test data to check for intrusions. It produces results used for intrusion detection.

3.1. Feature Engineering

It is the procedure for selecting the maximum -quality features to train the ML classifier, calculating the significance of each characteristic in class prediction, and selecting each feature. Figure 2: General feature selection technique based on filters

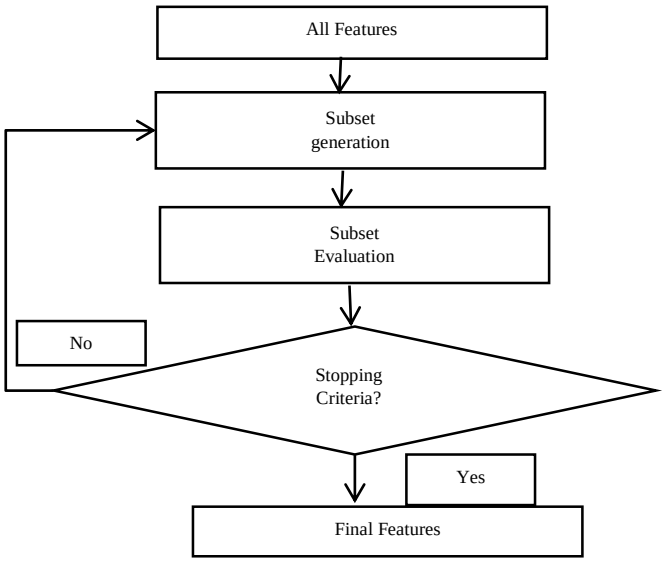

Fig. 2 Illustrates the feature selection process

In this manner, a metric determines feature relevance, and a filter-based method selects essential characteristics according to the feature-importance cutoff level. This study uses three measures to select attributes. This combines Entropy, Fisher, and the t-test criterion filtering methods. The available mean function of μ1(i) and σi(i) is the count of trends in both the unitary and null classes, n1 and n0. The KL-range Distance probability distribution uses Kullback-Leibler-the target's probability distribution. The Fisher index computation approach outlined in [36] is commonly used during the feature selection phase-Equation (1): Fisher Score.

$$F(i) = \left|\frac{\mu_1(i)-\mu_0(i)}{\sigma_1^2(i)-\sigma_0^2(i)}\right| \quad (1)$$

The figure index weights every feature or variable associated with the underlying data set. Another standard filter method, as discussed in [37], is the t-test. The computation is given in Equation (2): the t-score, and it is used to evaluate each feature's relative importance.

$$t(i) = \left|\frac{\mu_1(i)-\mu_0(i)}{\sqrt{\frac{\sigma_1^2(i)}{n_1}+\frac{\sigma_0^2(i)}{n_0}}}\right| \quad (2)$$

Relative entropy is just another standard filtering method. It is also known as the KL divergence. You may discover it in [38]. It calculates how two probabilistic distributions differ from one another-Equation (3): KL Divergence.

$$\mathrm{KL(p,q)}=\sum_i p_i log_2(\frac{p_i}{q_i}) \quad (3)$$

The proposed hybrid method utilises these three measurements to choose the optimal features relevant to this study.

Algorithm 1. Hybrid Feature Selection

Algorithm 1: Selection of Hybrid Features Selection (HFS)

Input:

- Dataset DD (CICIDS2017)
- Threshold th

Output:

- Selected features F

1. F ←Extract all features from D
2. For each feature $f \in F$
   a. Compute Fisher Score for f using Equation (1).
   b. Compute t-score for f Using Equation (2).
   c. Compute relative entropy for f Using Equation (3).
   d. Calculate the mean score of the metrics a, b, c
   e. Store the mean score of f in a map M.
3. F ←Select features based on the using MF \gets \text {Select features based on} th \text{using} M.
4. Return F.

An overview of Algorithm 1 is shown, which outputs the chosen features after receiving the inputs of dataset D and threshold th. The algorithm goes through numerous stages. These procedures are as follows: identify all attributes, create the entire feature space by considering every feature from every attribute, and then choose features using a hybrid filter and a threshold. Using a threshold value of 0.32 can get the best results in classifying the intrusion with the proposed algorithm.

3.2. Hyperparameter Tuning

The tuning process involves running algorithms like XGBoost, ExtraTrees, DecisionTree, and Random Forest on a schedule of gigabytes of data. Hyperparameter optimisation is essential. For XGBoost, parameters like learning rate, number of estimators, max tree depth, and feature and data sampling rates are tuned to attain optimal precision while balancing bias and variance. The ExtraTrees model exploits ensemble diversity and predictive power by optimising the number of estimators, feature subsets, and tree depth. Decision Tree models are tuned on the minimum samples for each split, minimum samples for each leaf, and tree depth, ensuring overfitting prevention and simplicity.

On the other hand, random forest tuning deals with the number of estimators, feature sampling, tree depth, and splitting criteria, and it uses ensemble learning. For all models, techniques like GS, RS, and BO are used to discover these hyperparameters - but each of these hyperparameter exploration methods can be very expensive.

3.3. ML Techniques

As illustrated in Figure 3, the intrusion detection procedure of the proposed system includes four ML models. All the models are independent of each other.

3.3.1. Decision Tree (DT)

The DT is an effective supervised learning technique it is applicable to duties involving both classification and regression. It creates what seems like a flowchart tree structure, with a class name at every leaf, an attribute test at each internal node, and a test result as directed branches. Until the base case-the tree's maximum depth-or the fewest samples a leaf may include before splitting-is reached, a recursive function separates the training data into subsets according to the attribute values.

In the training phase, the DT approach establishes which feature to use to Partition the data, such as entropy or Gini impurity, by calculating a metric that shows how pure or ordered the subsets are. The strategy is to find the characteristic that provides the most information or the most significant decrease in pollutants after the breakup.

3.3.2. Random Forest (RF)

The RF algorithm is a tree-learning ensemble ML technique. During training, a large number of DT are produced. Every tree is constructed with a subset of the features chosen at random. It introduces randomness that gives variance to the decision trees; overall accuracy is boosted, reducing the chances of overfitting.

The algorithm computes the average, or a vote, of each tree's results to make predictions. This collaborative approach to decision-making yields accurate results as trees and their perspectives assist. Random forests often handle regression and classification problems because they work well with high-dimensional data, reduce overfitting, and make accurate predictions.

3.3.3. Extra Trees (ET)

Many DTs are created using Additional trees method, similar to the random forest strategy, but each tree is replaced randomly. As a result, each tree has a unique sample from the dataset. Additionally, each tree includes a subset of features chosen randomly from the entire feature set. The most significant feature is also distinct for additional trees; this entails selecting a feature's splitting value at random. The method does not use entropy or Gini to help generate a locally optimal value. As a result, the trees become more diverse and interact.

3.3.4. XGBoost (XGB)

XGBoost is a scalable and efficient distributed gradient boosting toolbox that are suitable for training machine learning models. Several weak models' predictions are combined in ensemble learning to get a stronger prediction. XGBoost, also known as "extreme gradient boosting," is a well-liked general-purpose machine learning technique that has grown in popularity recently due to the fact that it can handle large datasets and produce innovative results for a range of ML problems, including regression and classification. XGBoost is a real-world, missing-value-free tool that can handle missing values. This function does not require much processing in advance. XGBoost can also utilise the inherent parallel processing capability of trees1, which allows for fast model training on large data sets.

3.4. Intrusion Detection Techniques

Two different kinds of ID techniques are used in the experiment design. The first involves classifying test samples into BENIGN and INTRUSION using a binomial classifier. The second tactic uses multi-class categorisation. Various groups indicate various invasion types. For Intrusion Class: BENIGN 0. The class index is used to assign the class to specific numeric ranges: DoS 1, Infiltration 2, PortScan 3, WebAttack 4, BruteForce 5, and Bot 6. The multiclass classification consists of these classes. Machine learning techniques are employed in this study for both multi-class and binomial classification.

Section 5 covers the implementation specifics. Python-based intrusion detection system implementation. SMOTE is used to overcome class imbalance and is only applied to the training partition after the split. Rebalance by filtering the minority classes by a threshold and sampling the majority-class data. How to create clusters out of data samples? K-Means grouping. Results generated during implementation include each model's confusion matrix and testing methods for both multi-class and binomial models.

Algorithm 2. Learning-Based Intrusion Detection (LbID)

Algorithm 2: Intrusion Detection based on Learning (LbID)

Input:

- Dataset D (CICIDS2017)
- Set of machine learning models M

Output:

- Intrusion detection results R
- Performance statistics P

1. Preprocess Data:
   $D' \leftarrow$ Preprocess D
2. Prepare Data:
   $(T_1, T_2) \leftarrow$ Split D' into training set T_1 and test set T_2

3. Feature Selection:
   F_1←Run the HFS Algorithm on T_1
   F_2←Retain features of F_1 in T_2
4. Model Training:
   For each model $m \in M$
   a. Train m using F_1.
   b. Save the trained model m.
   End For
5. Intrusion Detection and Evaluation:
   For each model $m \in M$:
   a. R ←Detect intrusions on F_2 using m.
   b. P←Evaluate performance of R Against ground truth.
   c. Print R .
   d. Print P.
   End For

The inputs to this method are the dataset and the set of ML models, as detailed in Algorithm 2. The algorithm provides preprocessing and data preparation so that the training and test data output will be T1 (training data) and T2 (test data), respectively. It trains each model iteratively with its respective features. This algorithm reuses the paper's suggested algorithm for choosing features on T1 only to obtain the ideal feature subset, which is then applied unchanged to T2. The models are trained on the particular feature being extracted during training. It predicts all corresponding intrusions when using the test data during its testing. The effectiveness of each ML model was evaluated by comparing the algorithm predictions against the ground truth values.

3.5. Evaluation Methodology

Using several standard metrics cited in the literature, this evaluates the suggested technique's performance as an IDS in trials.

These metrics are called F1-score, recall, accuracy, and precision. Figure 3 confusion matrix is used to calculate these.

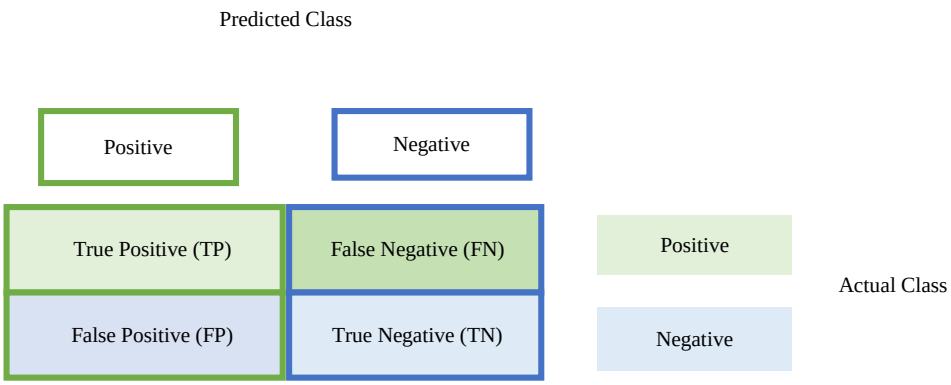

Fig. 3 Confusion matrix

If and only If the test sample provided contains an incursion both the algorithm and predicts it, the situation is termed a True Positive (TP) situation. This circumstance occurs when the test sample provided as well as the prediction algorithm are both benign. This is called a TN.

False Positive (FP): When a BENIGN test sample is provided, an algorithm predicts intrusion. False Negative (FN): This indicates that the test sample displayed INTRUSION even if the algorithm predicts BENIGN.

Equation (4): Precision: (p) = $\frac{TP}{TP+FP}$ (4)

Equation (5): Recall :(r) = $\frac{TP}{TP+FN}$ (5)

Equation (6): F1-score : (f1)= $2 * \frac{(p* r)}{(p+r)}$ (6)

Equation (7): Accuracy: = $\frac{TP+TN}{TP+TN+FP+FN}$ (7)

Precision, recall, F1-score and accuracy are calculated by Equations (4)-(7), respectively, based on the four instances in the confusion matrix. In case of multiple classes, precision and recall are computed as macro-average, and F1-score computed from them by applying Equation (6).

4. Experimental Outcomes

This assesses the suggested IDS by comparing the effectiveness of various machine learning models with and without optimisations. Additionally, the evaluation uses binomial and multi-class classification to test the analytical results. These are the study's findings.

Three categories are presented: binomial classification and two types of multiclass classification: one with optimisation and the other without. Patch via Gong et al. and the CICIDS2017 dataset [35] are used in this investigation.

Table 1. Performance of models using binomial classification for intrusion detection

Classification based on Binomial				
Models for Intrusion Detection	**Precision**	**Recall**	**F1-score**	**Accuracy**
XGBoost	83.37	87.89	85.57	89.27
ExtraTrees	89.89	93.54	91.68	93.54
DecisionTree	91.34	93.93	92.62	93.93
Random Forest	92.78	94.22	93.49	94.22

Table 1 shows each individual's performance detection model, including additional metrics such as precision. Binary accuracy ranged from 89.27% (XGBoost) to 94.22% (Random Forest).

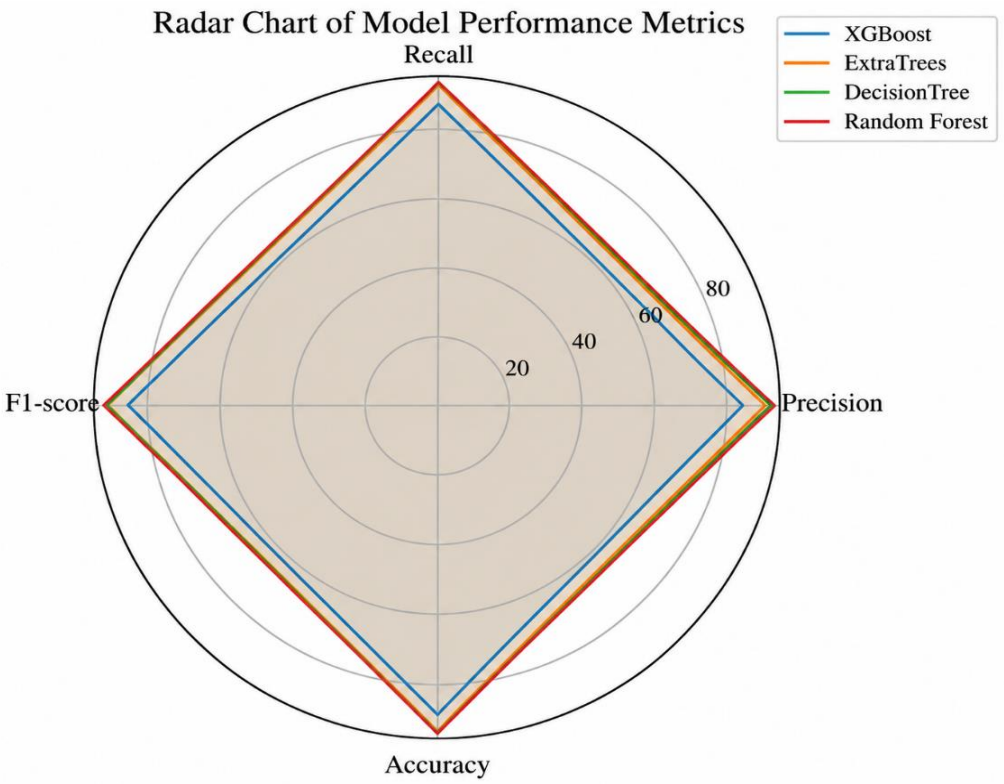

Fig. 4 Radar chart of model performance metrics

Figure 4: Radar chart comparing the performance of various ID models based on four key evaluation metrics: precision, recall, F1-score, and accuracy. With every model being a polygon in this view, you can see multiple measures at once for thousands of models together. The radar plot emphasises the balance and consistency of each model and highlights the models that perform well across all metrics. This visualisation gives a comprehensive insight into model effectiveness compared to traditional bar charts, and it also shows that ensemble-based approaches (Random Forest in particular) are the best and most balanced performing models overall across all evaluation criteria (4.2 Multi-Class Classification Outcomes without Optimisation).

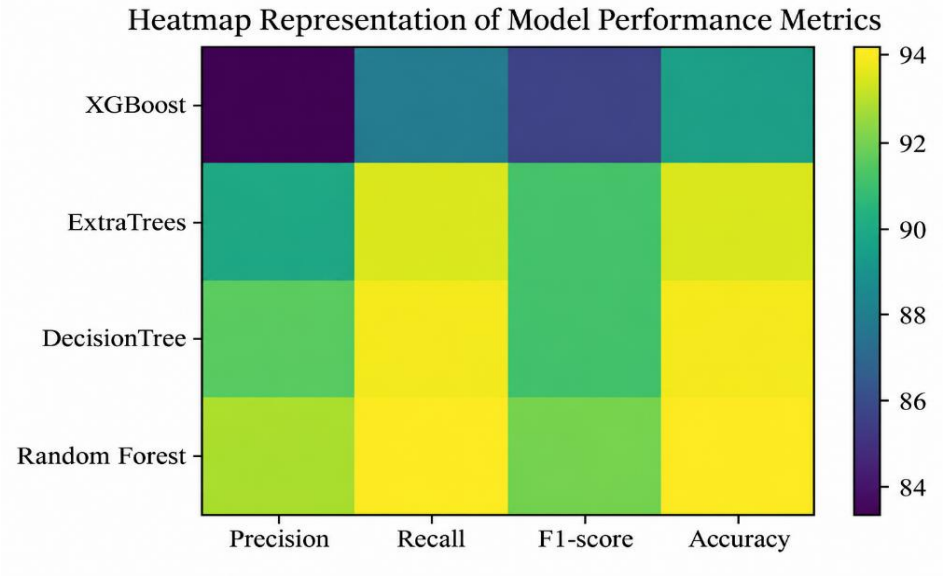

Fig. 5 Heatmap representation of model performance metrics

In Figure 5, the heatmap visualisation gives you an aggregated and intuitive insight into the model performance across a range of evaluation metrics. Every cell in the table corresponds to a metric value, and the gradient colour is directly correlated to the magnitude of performance. So, you can easily see which models and metrics have strong or weak visual patterns.

The above heatmap clearly illustrates that Random Forest and Decision Tree models yield higher values across most of the metrics, whereas XGBoost is by far the lowest performer. This visualisation also facilitates interpretability by translating numbers into colours, making it easier to compare results on the graph than to compare the numerical results directly.

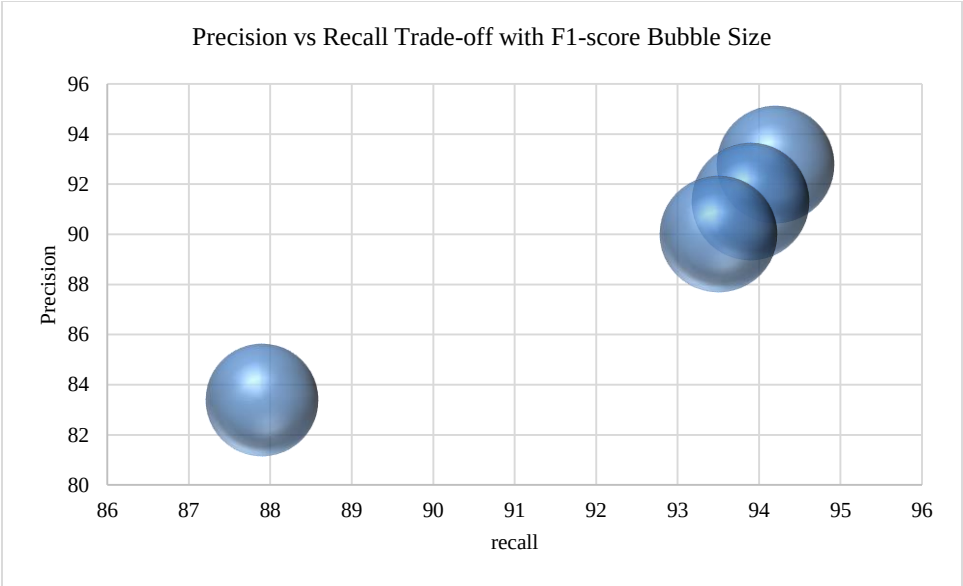

Fig. 6 Precision vs recall trade-off with F1-score bubble size

Figure 6 shows the compromise between recall and precision for each model (scatter plot) and indicates the F1-score with the bubble size. This cube can be visualised as multiple dimensions to analyse the trade-off between correctly identifying positive instances and minimising false positives. Models located towards the top-right represent the best performance. Bubble size is a further indication of this harmonic balance between Precision and Recall. The plot indicates that Random Forest balanced well in F1, with high precision and recall, providing the best model compared to the other models tried in this work.

Table 2. Multi-Class classification models' performance for intrusion detection without optimisation

Multi-Class Classification Performance (NOT Optimised)				
Model of Intrusion Detection	**Precision**	**Recall**	**F1-score**	**Accuracy**
Random Forest	76.61	92.06	83.63	88.62
XGBoost	84.99	80.76	82.82	90.54
ExtraTrees	89.42	91.35	90.37	90.89
DecisionTree	92.78	94.61	93.69	91.67

Table 2 presents each detection model's performance outcomes for categorization across classes without tuning on precision as well as other measures. Multi-class accuracy ranged from 88.62% (Random Forest) to 91.67% (DecisionTree).

As shown in Figure 7, a distribution-based visualisation of performance metrics (Box-style spread plot) shows the spread of Precision, Recall, F1-score, and Accuracy values for each model. This visualisation shows both central tendency and spread of model performance.

Tight distributions show that the metrics are performing similarly, and a wider distribution indicates that performance is more variable. It provides deeper insight into model stability, as shown through the range plots, which is very useful for conducting robustness tests in multi-class intrusion detection systems.

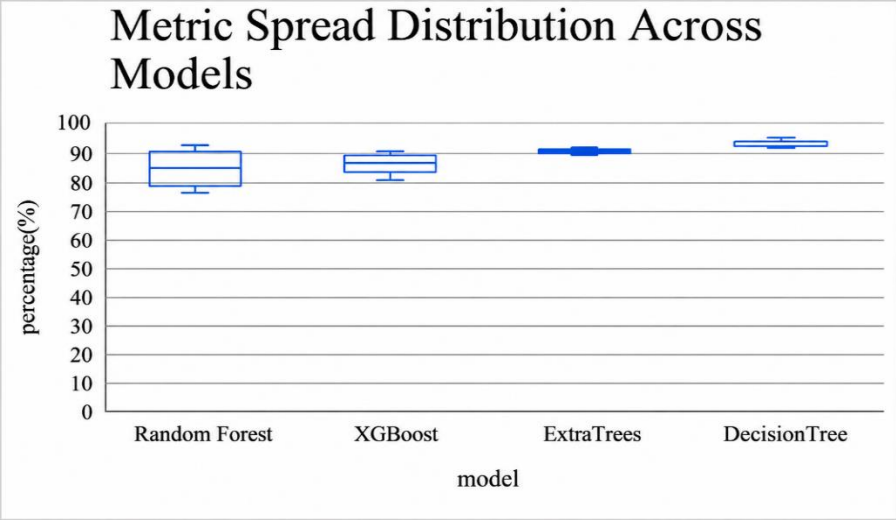

Fig. 7 Metric spread distribution across models

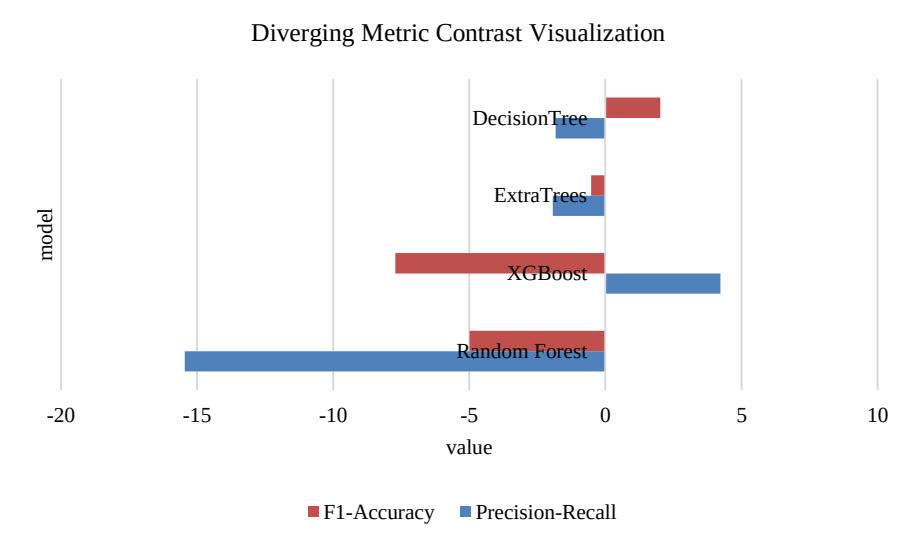

Fig. 8 Diverging metric contrast visualisation

Figure 8 shows the differences in some key performance metrics (displayed using a diverging bar plot, and centred around zero). If the values are positive, then Precision is better than Recall (or F1 is better than Accuracy), and if the values are negative, then it is the other way around. This abstraction cleanly emphasises imbalances in a metric and performance in one direction vs another, for each model. It also helps quickly identify models that are favouring given metrics, enabling a better understanding of model behaviour. So, moving forward, it seems a better option for interpretability because it focuses on relative distance rather than the absolute.

This 2D Figure 9 efficiency plot visualises model accuracy versus a metric variance represented as flat surfaces plotted on top of each other. Each point represents a model, and together, you can see the performance vs. stability trade-off line. The models in the lower-right corner are ideal for higher accuracy and lower variability. By using both effectiveness and consistency to create this quadrant-based visualisation, this provides a more intuitive framework for model selection. It is broader than the one-parameter-at-a-time assessment true for most traditional comparisons.

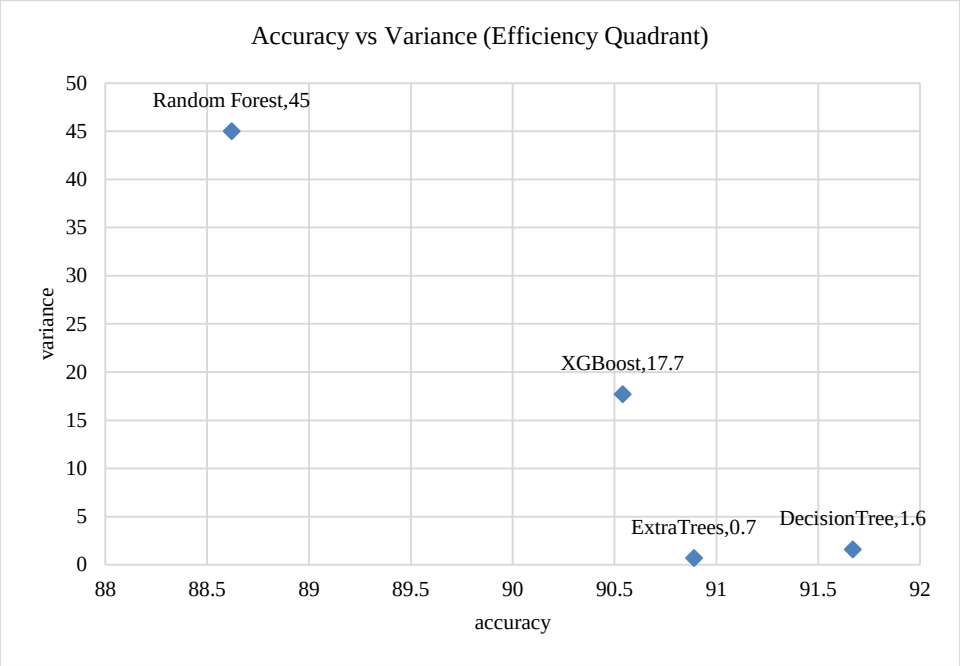

Fig. 9 Accuracy vs variance (Efficiency quadrant)

Table 3. Performance of models with multi-class classification that are optimised for intrusion detection

Performance of multi-class classification (With optimisation)				
Models of Intrusion Detection	**Precision**	**Recall**	**F1-score**	**Accuracy**
DecisionTree	84.99	90.26	87.55	84.36
ExtraTrees	76.61	84.85	80.52	89.37
XGBoost	88.64	85.61	87.10	91.34
Random Forest	83.37	87.89	85.57	93.46

Table 3 shows the accuracy and other parameters of multi-class classification for each detection model Optimisation – with optimisation (HFS and hyperparameter tuning). The accuracy of the models was between 84.36% (DecisionTree) and 93.46% (Random Forest).

Optimisation resulted in an increment of +4.84 Percentage Points (PCP) for Random Forest model, +0.80 PCP for XGBoost model, while DecisionTree model and ExtraTrees model had decrements of -7.31 PCP and -1.52 PCP, respectively. Hence, the benefit of optimisation is model-dependent.

Figure 10 shows the Performance Balance Index (PBI), which is computed using the standard deviation of many key performance indicators, including Accuracy, Precision, Recall, and F1 Score. A smaller PBI means performance across all metrics is more uniform and balanced. Instead of the single metrics typically used to evaluate each model, this visualisation shows the stability of each model. Such lower-variability models reflect robustness and reliability for multi-class classification. This ideal plot indicates that having a balanced distribution across all metrics is a more appropriate model for practical IIDSs, where balance across several evaluation criteria is critical.

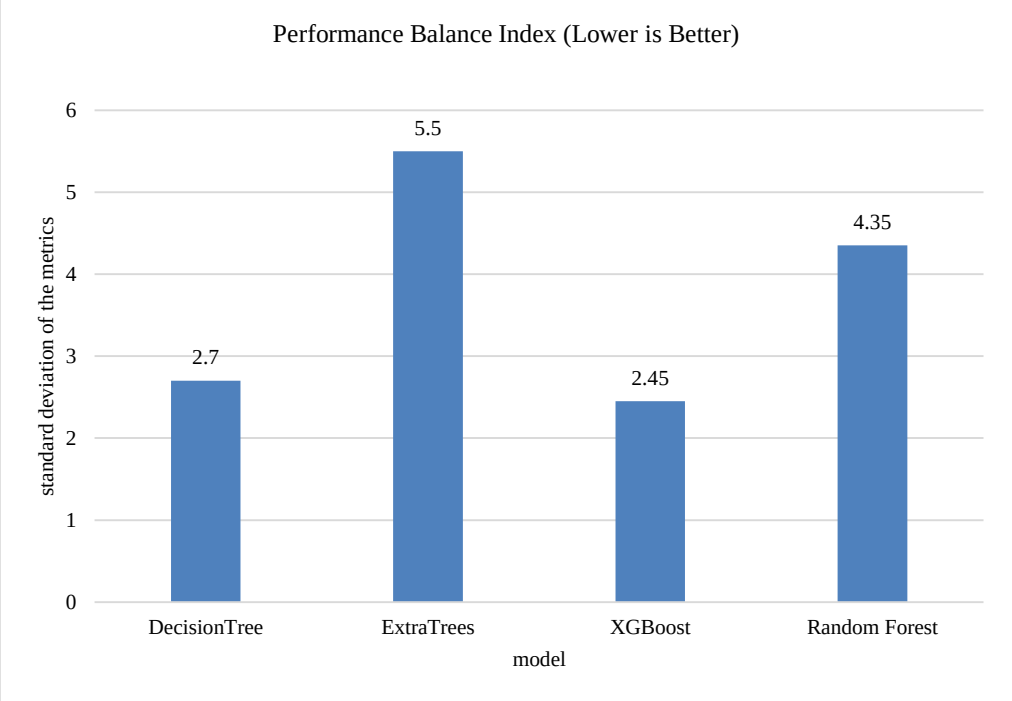

Fig. 10 Performance balance index (Lower is better)

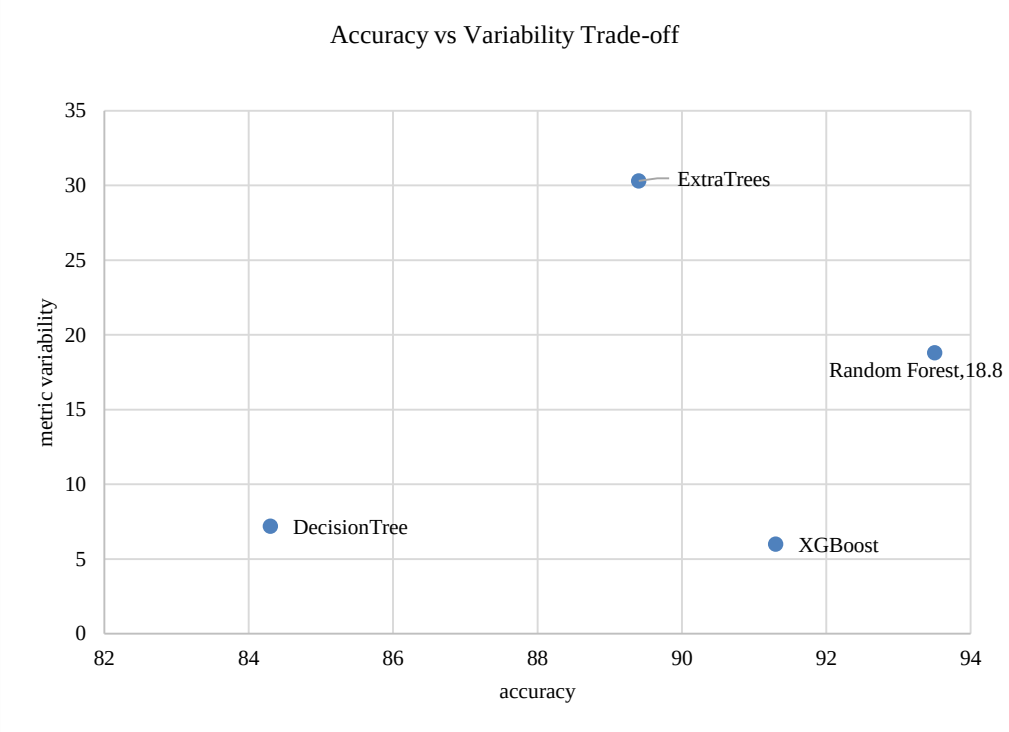

Fig. 11 Accuracy vs variability trade-off

Figure 11 visualises the relationship between accuracy and variability of performance metrics. Each dot corresponds to a model, and the X-axis indicates its accuracy, while the Y-axis indicates its variance (across Precision, Recall and F1-score). This visualisation shows the balance between high accuracy and good performance across metrics. Models closer to the lower-right quadrant perform the best-they yield high accuracy with low variability. This plot provides more context for model reliability; it allows the selection of a model that performs well but is also stable across evaluation measures.

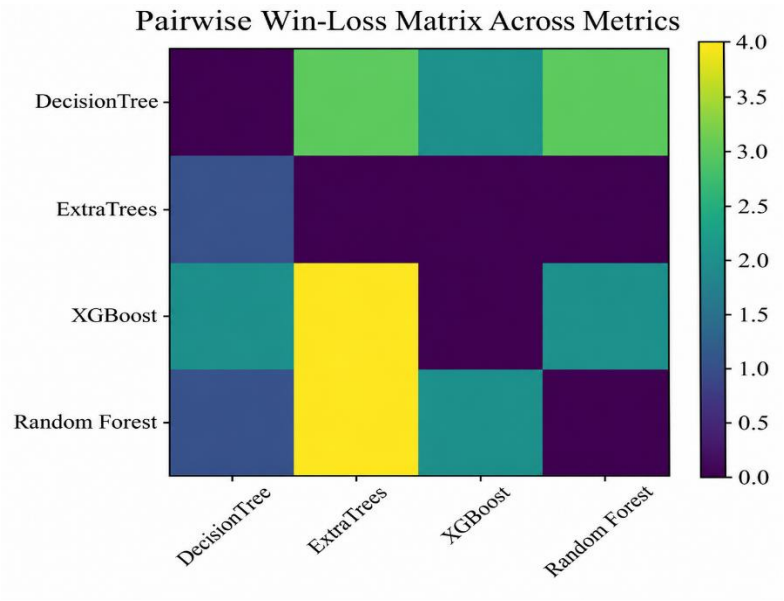

Fig. 12 Pairwise win-loss matrix across metrics

A win-loss matrix comparing pairs of models across various evaluation metrics is illustrated in Figure 12. The cells show how many metrics one model performs better than the other. Values higher than 0 indicate the extent to which one model dominates over the other. This allows you to compare individual metrics, but this visualisation also provides a broad path to compare models against each other, where you can even compare across metrics to highlight how they are comparatively good and bad! The main advantage is that it allows you to identify the best-performing models and provide a better picture of their competitiveness when applied to multi-class intrusion detection problems.

Table 4. Effect of optimisation on random forest performance

Random Forest Configuration	Accuracy (%)	Absolute Improvement (percentage points)
Without Optimization	88.62	—
With HFS and Hyperparameter Optimisation	**93.46**	**4.84**

The results of the proposed optimisation strategy on the performance of Random Forest are shown in Table 4.

The base model Random Forest had an accuracy of 88.62%, and the optimised model had an accuracy of 93.46%, with a positive difference of 4.84 percentage points.

This improvement shows that combining feature selection and hyperparameter optimisation can significantly enhance the multiclass intrusion-detection ability of the Random Forest classifier. This outcome also demonstrates that the obtained Execution is not entirely the impact of the given classifiers, but is also related to model optimisation and feature representation.

Table 5. Comparison of the proposed model with existing intrusion detection methods

Ref. / Author	Proposed Method	Optimisation / Feature Engineering	Accuracy (%)	Improvement of Proposed Model (percentage points)
Gao et al. [38]	Ensemble ML: DT + RF + KNN + DNN + MultiTree	PCA + weighted majority voting	85.2	**8.26**
Wei et al. [39]	Deep Belief Network (DBN)	PSO + AFSA + GA	82.36	**11.1**
Ali et al. [40]	Fast Learning Network (FLN)	Particle Swarm Optimisation (PSO)	89.23	**4.23**
Shetty [41]	Random Forest + SHAP	SHAP-based explainable AI	92	**1.46**
Proposed Model	Optimised Random Forest	Hybrid Feature Selection + Hyperparameter Optimisation	93.46	—

Table 5 demonstrates the outcomes of the proposed optimised RF comparison of the model to representative intrusion-detection models reported in the literature. Using an adaptive ensemble of multiple machine-learning classifiers, Gao et al. [38] achieved 85.20% accuracy, and using an AFSA-GA-PSO-optimized Deep Belief Network, Wei et al. [39] achieved 82.36% accuracy. Ali et al. [40] designed an optimised Fast Learning Network using the PSO technique and achieved an overall accuracy of 89.23%. Shetty [41] used Random Forest coupled with an explainable approach using the SHAP method in CICIDS2017 and achieved 92% accuracy between benign and malicious traffic using this method. On the contrary, the optimum Random Forest achieved 93.46% accuracy by using HFS and hyperparameter tuning. From the outcomes, it is evident that the suggested framework shows higher reported accuracy than the selected approaches. This improvement is attributed to the collective optimisation of the feature space and the classifier configuration parameters: the informative traffic features selected by Hybrid Feature Selection and the hyperparameter optimisation of the Random Forest model from the selected features.

5. Discussion

This research work describes the mechanism of detecting intrusions with ML models being trained, developed, and deployed. It relies upon supervised learning because tagged data are available for accurate observation. This study uses the CICIDS2017 dataset. Research on datasets indicated that, while older and more cited than other datasets, it outperformed many other datasets on benchmarking and intrusion type support [2]. Though many machine learning models exist, four tree-based techniques are chosen in this study. This is justified by tree-based models performing better based on what was discussed in Section 2: a literature survey. Remember that because the selected ML models are supervised, the quality of the training data is crucial. As a result, we use the proposed HFS (Fisher score, t-test and KL divergence) to enhance the quality of the training data. Feature selection was one of the optimisation techniques applied (among others). Hyperparameter tuning via grid search, random search, and Bayesian optimisation (Section 3.2) is the second method of optimisation.

The experiments are divided into three categories. The first set of trials employs a binomial classification strategy using machine learning models. Each model, such as INTRUSION and BENIGN, separates test samples into two groups. The second set of experiments uses multi-class classification-oriented models that do not include the adjustments above. Accordingly, the test samples will be separated into multiple groups by each model, such as Bot, BruteForce, DoS, Infiltration, PortScan, and WebAttack. Trials in the third category use Multi-Class Classification with the aforementioned enhancements, and the ML models are also rooted in this category. This implies which every model will categorise test samples and group them into various classes, including WebAttack, BruteForce, DenialOfService, Infiltration, PortScan, and Bot. It is recommended to use the first category since it provides output for every test case, either BENIGN or INTRUSION. This converts well here, as for many use cases, categorisation is unnecessary, and intrusion detection is enough. In several use cases, the network administrator is interested in the exact nature of the intrusion or attack. In these scenarios, multi-class categorisation is functional. These might increase performance, but writing such a statement could raise eyebrows, as long as the identified research gaps state that improving ML models significantly depends on feature engineering and hyperparameter optimisation. Methods centred on these have also reported a lack of accuracy (despite the few that did), along lines similar to those pointed out in the literature.

6. Conclusion and Future Scope

This project sought to build and implement ML-centric IDSs using four benchmark tree–based models: Random Forests (RF), Extra Trees (ETs), Decision Tree (DT), and XGBoost. In addition, the framework applied two optimisation techniques, namely feature selection and hyperparameter tuning, along with the proposed Learning-based Intrusion Detection (LbID) and Hybrid Feature Selection (HFS) algorithms. The optimisation techniques These had model-dependent effects on accuracy (+4.84 and +0.80 percentage points for RF and XGBoost, respectively, while DT and ET decreased), meaning that these gains might be constrained by the data. Based on the experimental results, the RF model achieved the highest accuracy for binomial classification (94.22%), and the DT provided the best overall accuracy for multi-class classification (91.67%) when not optimised. The Random Forest classifier, when tuned, outperformed the other models in multi-class classification with 93.46% accuracy.

Nevertheless, the dependence on the very old CICIDS2017 dataset limits generalisation and the ability to mitigate new attacks. This represents an important area for improvement for subsequent studies. Building on this work, future research should focus on more recent and active datasets such as UNSW-NB15, CSE-CIC-IDS2018, or BoT-IoT, which represent more challenging and realistic cyber threats, especially in environments driven by IoT technology. In addition, DL architectures like CNNs, RNNs, and Transformer-based models can reveal temporal and spatial information in network traffic data. Models like these could vastly exceed the precision and robustness to new threats of traditional algorithms. Furthermore, this system could be made live with an intrusion detection system fed through streaming data frameworks like Apache Kafka or Spark Streaming and proactively mitigated. To improve the detection process, innovations like autoencoders for unsupervised anomaly detection, GANs for synthetic attack data generation, and attention-based mechanisms to model

inter-feature correlations can also be incorporated into the system. Finally, this can move from simulated datasets to live traffic data and evaluate the real-world effectiveness of ADs, while introducing additional checks to make intrusion detection systems deployment-ready.

Though the current research investigated only classical ML models, future work could implement DL architectures such as CNN, RNN, and Transformer architectures to allow more effective and adaptable intrusion detection. Integrating these models with real-time or streaming data platforms, such as Apache Kafka or Spark Streaming, could also enable the development of real-time detection systems that process live network traffic, thus enabling proactive threat mitigation in dynamic enterprise environments. While static and outdated datasets like CICIDS2017 still play a role in comparisons and performance evaluations of legacy systems, future work could benefit from using more modern and realistic datasets such as NSL-KDD, UNSW-NB15, and CSE-CIC-IDS2018 for modern attack types, and the TON_IoT or BoT-IoT dataset collections for IoT-specific threats. Furthermore, techniques such as GANs could advance the methodological approach by using augmented artificial data, autoencoders to detect anomalous or malicious activity, and attention mechanisms to model dependencies between features, which may help improve detection accuracy, adaptability, and generalizability across different networks.

Reference

[1] Chie-Hong Lee et al., "Machine Learning based Network Intrusion Detection," *2017 2nd IEEE International Conference on Computational Intelligence and Applications*, Beijing, China, pp. 79-83, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[2] Razan Abdulhammed et al., "Features Dimensionality Reduction Approaches for Machine Learning based Network Intrusion Detection," *Electronics*, vol. 8, no. 3, pp. 1-27, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[3] Elie Alhajjar, Paul Maxwell, and Nathaniel Bastian, "Adversarial Machine Learning in Network Intrusion Detection Systems," *Expert Systems with Applications*, vol. 186, pp. 1-13, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[4] Gustavo De Carvalho Bertoli et al., "An End-to-End Framework for Machine Learning-based Network Intrusion Detection System," *IEEE Access*, vol. 9, pp. 106790-106805, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[5] Zeeshan Ahmad et al., "Network Intrusion Detection System: A Systematic Study of Machine Learning and Deep Learning Approaches," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 1, pp. 1-29, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[6] Marzia Zaman, and Chung-Horng Lung, "Evaluation of Machine Learning Techniques for Network Intrusion Detection," *2018 IEEE/IFIP Network Operations and Management Symposium*, Taipei, Taiwan, pp. 1-5, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[7] Nasrin Sultana et al., "Survey on SDN based Network Intrusion Detection System using Machine Learning Approaches," *Peer-to-Peer Networking and Applications*, vol. 12, no. 2, pp. 493-501, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[8] Jie Li et al., *Machine Learning Algorithms for Network Intrusion Detection*, Springer, vol. 151, pp. 151-179, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[9] Anshu Parashar, Kuljot Singh Saggu, and Anupam Garg, "Machine Learning based Framework for Network Intrusion Detection System using Stacking Ensemble Technique," *Indian Journal of Engineering and Materials Sciences*, vol. 29, no. 4, pp. 509-518, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[10] Kazi Abu Taher, Billal Mohammed Yasin Jisan, and Md. Mahbubur Rahman, "Network Intrusion Detection using Supervised Machine Learning Technique with Feature Selection," *2019 International Conference on Robotics, Electrical and Signal Processing Techniques*, Dhaka, Bangladesh, pp. 643-646, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[11] Roberto Magán-Carrión et al., "Towards a Reliable Comparison and Evaluation of Network Intrusion Detection Systems based on Machine Learning Approaches," *Applied Sciences*, vol. 10, no. 5, pp. 1-21, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[12] Anish Halimaa A., and K. Sundarakantham, "Machine Learning based Intrusion Detection System," *2019 3rd International Conference on Trends in Electronics and Informatics*, Tirunelveli, India, pp. 916-920, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[13] Aditya Phadke et al., "A Review of Machine Learning Methodologies for Network Intrusion Detection," *2019 3rd International Conference on Computing Methodologies and Communication*, Erode, India, pp. 272-275, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[14] Pierpaolo Dini, and Sergio Saponara, "Analysis, Design, and Comparison of Machine-Learning Techniques for Networking Intrusion Detection," *Designs*, vol. 5, no. 1, pp. 1-22, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[15] Preeti Mishra et al., "A Detailed Investigation and Analysis of using Machine Learning Techniques for Intrusion Detection," *IEEE Communications Surveys and Tutorials*, vol. 21, no. 1, pp. 686-728, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[16] Kelton A.P. da Costa et al., "Internet of Things: A Survey on Machine Learning-based Intrusion Detection Approaches," *Computer Networks*, vol. 151, pp. 147-157, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[17] B. Ida Seraphim et al., "Survey on Machine Learning Techniques in Network Intrusion Detection System," *2018 4th International Conference on Computing Communication and Automation*, Greater Noida, India, pp. 1-5, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[18] T. Rupa Devi, and Srinivasu Badugu, "A Review on Network Intrusion Detection System using Machine Learning," *International Conference on E-Business and Telecommunications*, Prague, Czech Republic, pp. 598-607, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[19] Mohammad Almseidin et al., "Evaluation of Machine Learning Algorithms for Intrusion Detection System," *2017 IEEE 15th International Symposium on Intelligent Systems and Informatics*, Subotica, Serbia, pp. 000277-000282, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[20] Lan Liu et al., "Intrusion Detection of Imbalanced Network Traffic based on Machine Learning and Deep Learning," *IEEE Access*, vol. 9, pp. 7550-7563, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[21] Fei Wu et al., "Research on Network Intrusion Detection Technology based on Machine Learning," *International Journal of Wireless Information Networks*, vol. 28, no. 3, pp. 262-275, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[22] Achmad Akbar Megantara, and Tohari Ahmad, "A Hybrid Machine Learning Method for Increasing the Performance of Network Intrusion Detection Systems," *Journal of Big Data*, vol. 8, no. 1, pp. 1-19, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[23] Yaping Chang, Wei Li, and Zhongming Yang, "Network Intrusion Detection based on Random Forest and Support Vector Machine," *2017 IEEE International Conference on Computational Science and Engineering and IEEE International Conference on Embedded and Ubiquitous Computing*, Guangzhou, China, pp. 635-638, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[24] Atiku Abubakar, and Bernardi Pranggono, "Machine Learning based Intrusion Detection System for Software Defined Networks," *2017 Seventh International Conference on Emerging Security Technologies*, Canterbury, UK, pp. 138-143, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[25] Farrukh Aslam Khan, and Abdu Gumaei, "A Comparative Study of Machine Learning Classifiers for Network Intrusion Detection," *International Conference on Artificial Intelligence and Security*, Springer, Cham, pp. 75-86, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[26] N. Thomas Rincy, and Roopam Gupta, "Design and Development of an Efficient Network Intrusion Detection System using Machine Learning Techniques," *Wireless Communications and Mobile Computing*, vol. 2021, no. 1, pp. 1-35, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[27] Jing Li et al., "Optimizing IoT Intrusion Detection System: Feature Selection versus Feature Extraction in Machine Learning," *Journal of Big Data*, vol. 11, no. 1, pp. 1-44, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[28] Ameera S. Jaradat, Malek M. Barhoush, and Rawan S. Bani Easa, "Network Intrusion Detection System: Machine Learning Approach," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 25, no. 2, pp. 1151-1158, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[29] Zhihui Fan, and Zhixuan You, "Research on Network Intrusion Detection based on XGBoost Algorithm and Multiple Machine Learning Algorithms," *Theoretical and Natural Science*, vol. 31, pp. 161-166, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[30] Chunying Zhang et al., "Three-Branch Random Forest Intrusion Detection Model," *Mathematics*, vol. 10, no. 23, pp. 1-21, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[31] Md. Alamin Talukder et al., "Machine Learning-based Network Intrusion Detection for Big and Imbalanced Data using Oversampling, Stacking Feature Embedding and Feature Extraction," *Journal of Big Data*, vol. 11, no. 1, pp. 1-44, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[32] Sukhpreet Singh Dhaliwal, Abdullah-Al Nahid, and Robert Abbas, "Effective Intrusion Detection System using XGBoost," *Information*, vol. 9, no. 7, pp. 1-24, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[33] Witcha Chimphlee, and Siriporn Chimphlee, "Intrusion Detection System (IDS) Development using Tree-based Machine Learning Algorithms," *International Journal of Computer Networks and Communications*, vol. 15, no. 4, pp. 93-109, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[34] Canadian Institute for Cybersecurity, Intrusion Detection Evaluation Dataset, Canadian Institute for Cybersecurity, 2017. [Online]. Available: https://www.unb.ca/cic/datasets/ids-2017.html

[35] Sebastián Maldonado, and Richard Weber, "A Wrapper Method for Feature Selection using Support Vector Machines," *Information Sciences*, vol. 179, no. 13, pp. 2208-2217, 2009. [CrossRef] [Google Scholar] [Publisher Link]

[36] John A. Rice, *Mathematical Statistics and Data Analysis*, China Machine Press, 2003. [Google Scholar]

[37] S. Kullback, and R.A. Leibler, "On Information and Sufficiency," *Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79-86, 1951. [Google Scholar] [Publisher Link]

[38] Xianwei Gao et al., "An Adaptive Ensemble Machine Learning Model for Intrusion Detection," *IEEE Access*, vol. 7, pp. 82512-82521, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[39] Peng Wei et al., "An Optimization Method for Intrusion Detection Classification Model based on Deep Belief Network," *IEEE Access*, vol. 7, pp. 87593-87605, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[40] Mohammed Hasan Ali et al., "A New Intrusion Detection System based on Fast Learning Network and Particle Swarm Optimization," *IEEE Access*, vol. 6, pp. 20255-20261, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[41] Ashwitha C Shetty, "Explainable Network Intrusion Detection using Random Forest and SHAP on the CICIDS2017 Dataset," *International Journal for Research in Applied Science and Engineering Technology*, vol. 14, no. 1, pp. 1883-1886, 2026. [CrossRef] [Publisher Link]