Hybrid Stacking Model for Web Attack Classification Using LightGBM, Random Forest, and MLP

Fadli Dony Pradana[1*], Farikhin[2], and Budi Warsito[3]
[1−3]Master Program of Information Systems, Postgraduate School, Universitas Diponegoro
Semarang, Indonesia 50241
Email: [1]fadlidonypradana@gmail.com, [2]farikhin.math.undip@gmail.com,
[3]budiwarsitoundip@gmail.com

***Abstract*—The research presents a stacking-based hybrid intrusion detection framework for web application attacks, addressing the persistent limitation that minority classes, including Brute Force, Cross-Site Scripting (XSS), and Structured Query Language (SQL) Injection, are frequently underdetected in conventional Intrusion Detection Systems (IDS) due to severe class imbalance. The proposed architecture combines LightGBM and Random Forest as base learners, while a Multi-Layer Perceptron (MLP) functions as the meta-learner. The framework is supported by rigorous preprocessing, ANOVA F-test-based feature selection, and domain-informed augmentation of critical traffic features, such as Flow Inter-Arrival Time (IAT) Min, Init_Win_bytes_forward, and Backward (Bwd) Packets/s, through optimized weighting strategies. Evaluation on the CICIDS-2017 web attack subset using 10-fold stratified cross-validation shows that the proposed model improves the macro F1-Score from 0.62 ± 0.004 to 0.76 ± 0.003 and achieves a binary accuracy of 99.67% with a macro F1 of 0.94. The observed performance gains are statistically significant ($p < 0.001$), confirming the robustness of the framework. These findings indicate that targeted feature engineering and heterogeneous stacking substantially improve minority-attack detection while preserving majority-class performance. In addition, the framework demonstrates sub-millisecond inference time, highlighting its practical suitability for real-time IDS deployment in resource-constrained and high-throughput operational cybersecurity environments. The proposed design also offers methodological generalizability for broader anomaly detection tasks in dynamic network environments, where reliable recognition of low-frequency but high-impact attack patterns remains increasingly critically important.**

***Index Terms*—Stacking Hybrid, Web Attack Classification, Intrusion Detection System (IDS), Light Gradient Boosting Machine (LightGBM), Random Forest, Multi-Layer Perceptron (MLP)**

Received: June 03, 2025; received in revised form: Sep. 23, 2025; accepted: Sep. 24, 2025; available online: April 02, 2026.
*Corresponding Author

I. Introduction

THE growing dependence on web applications has made people increasingly susceptible to advanced cyberattacks, jeopardizing data confidentiality, service availability, and financial viability. In response, Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS) have become the mainstay of network security. Signature-based IDSs are still commonly used because they are highly accurate in detecting known threats. However, they are ineffective against zero-day attacks. Since threats continuously evolve, they produce a high number of false negatives due to delays in signature updates [1, 2].

To address these limitations, Machine Learning (ML) methods have been widely adopted in IDS research to enhance adaptability. Classical algorithms, such as Random Forest, Naïve Bayes (NB), and K-Nearest Neighbors (KNN), are favored for their simplicity and low computational overhead. However, they exhibit poor generalization when detecting complex or novel attacks [3]. In contrast, deep learning models, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), can capture sequential dependencies and hidden attack patterns [4, 5]. Nevertheless, their high training costs and limited interpretability hinder their deployment in real-time applications [6, 7]. Consequently, achieving a practical balance between efficiency, detection accuracy, and explainability remains a persistent challenge in IDS research.

Recent literature indicates that deep learning models in IDS often suffer from the vanishing gradient problem, slow convergence, and sensitivity to hyperparameter settings [8]. As a result, hybrid and ensemble learning methods have emerged as promising alternatives to enhance detection performance through model diversity. Studies integrating CNNs, autoencoders, and

Decision Trees have demonstrated substantial improvements, achieving up to 95% accuracy on benchmark datasets such as CICIDS2017 [9]. Similarly, ensemble voting methods have shown notable performance gains [10].

Despite these advancements, several key challenges remain, including computational scalability and class imbalance [11]. Techniques, such as XGBoost-based feature selection and Long Short-Term Memory (LSTM) classifiers, have been employed to improve efficiency and feature compactness [12]. Transformer-based models, particularly when combined with Synthetic Minority Oversampling Technique (SMOTE) and evolutionary optimization, have reported high detection rates under skewed class distributions [13]. Moreover, semi-supervised approaches have been utilized to reduce false positives in low-labeled environments, thereby aligning more closely with real-world deployment constraints [14].

Despite progress in IDS research, web-specific attacks, such as Structured Query Language (SQL) Injection, Cross-Site Scripting (XSS), and Brute Force, remain underexplored and underrepresented, which makes accurate detection challenging. Recent ensemble and magnet loss approaches have improved discrimination capability and computational efficiency [15, 16]. However, issues related to sensitivity, interpretability, and response time persist.

Individual models show promise but involve trade-offs between detection quality and system cost. Shallow learners are fast and interpretable, but they fail to capture complex non-linear patterns. In contrast, deep models are powerful but computationally costly and less transparent. Stacking offers an alternative by integrating heterogeneous models through a meta-learner, thereby leveraging complementary strengths to improve IDS performance under class imbalance and evolving attacks. The concept of stacked generalization, along with Breiman's work on bagging and Random Forest, forms the theoretical foundation for such ensemble approaches [17–19].

A stacking-based intrusion detection architecture is introduced to classify web-based attacks by leveraging classifier diversity for robust prediction. The framework combines Light Gradient Boosting Machine (LightGBM) and Random Forest as base learners. LightGBM provides fast training and effectively handles varying feature magnitudes [20], while Random Forest offers robustness against noise and overfitting in high-dimensional data [21]. Their outputs are integrated using MLP meta-learner to enhance accuracy and generalization [22]. To further improve performance, the framework incorporates domain-specific feature augmentation, Analysis of Variance (ANOVA)-based feature selection, and hyperparameter tuning using Tree-structured Parzen Estimators (TPE). The model is evaluated using the CICIDS-2017 dataset, which has been extensively analyzed and recognized for including diverse and contemporary attack types, including web-based threats such as XSS, Brute Force, and SQL Injection, making it a suitable benchmark for assessing detection frameworks under class-imbalanced and realistic network conditions [23]. However, despite these advances, most ensemble-based IDS studies remain focused on generic intrusions and fail to address the distinct characteristics of web application attacks.

Consequently, three key research gaps persist: (i) insufficient attention to minority web-attack classes such as Brute Force, XSS, and SQL Injection; (ii) the absence of systematic integration between stacking architectures and domain-informed feature augmentation to mitigate skewed class distributions; and (iii) limited adoption of explainable AI mechanisms to improve model transparency and facilitate real-world deployment. Moreover, the research has three objectives: (i) to design a stacking-based IDS that prioritizes the detection of minority web-attack classes (Brute Force, XSS, and SQL Injection); (ii) to integrate domain-informed feature augmentation with a heterogeneous stacking architecture to mitigate class imbalance while preserving generalization; and (iii) to enhance the transparency of the meta-learner through SHapley Additive exPlanations (SHAP)-based explainability to facilitate real-world deployment.

II. Research Method

The proposed classification method employs a stacked hybrid model to identify web-based attacks in network traffic. The overall architecture of the proposed procedure is comprehensively illustrated in Fig. 1. It covers the entire process from data collection to final evaluation.

A. Data Collections

The research utilizes the CICIDS 2017 Web Attack subset, specifically the Thursday-WorkingHours-Morning-WebAttacks capture file. The dataset contains session-level traffic that simulates realistic web attacks such as SQL Injection, XSS, and Brute Force. It comprises 458,968 entries and 85 attributes (see Table I). However, only 170,366 rows contain complete data. Therefore, preprocessing is required to address missing values. The features include both categorical and numerical variables representing session duration, packet statistics, and flow behavior [24].

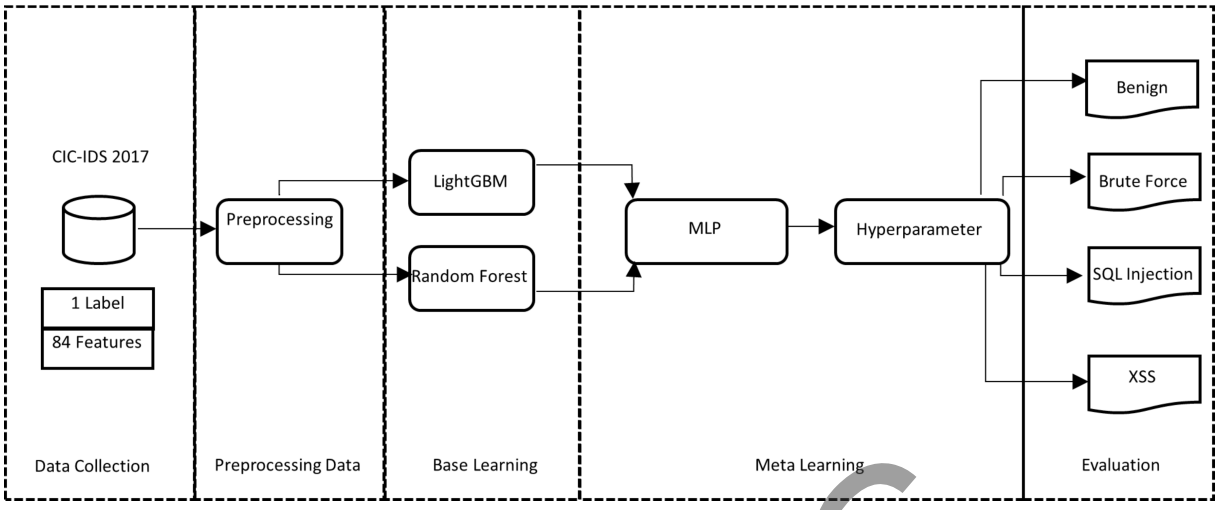

Fig. 1. Overview of The proposed stacked hybrid intrusion detection framework. Note: Light Gradient Boosting Machine (LightGBM), Multi-Layer Perceptron (MLP), Structured Query Language (SQL), and Cross-Site Scripting (XSS).

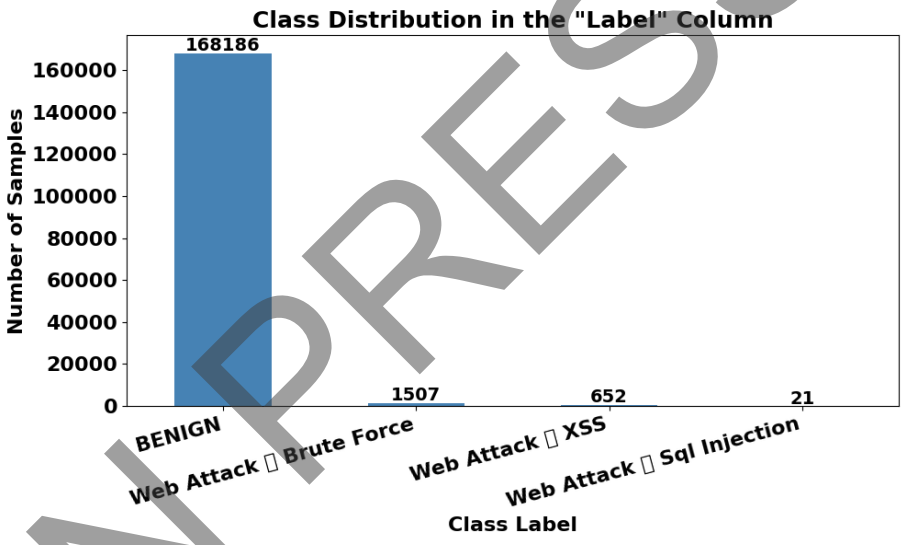

Fig. 2. Distribution of classes in the target label of the dataset. Note: Structured Query Language (SQL) and Cross-Site Scripting (XSS).

The label distribution is highly imbalanced, with BENIGN traffic significantly outnumbering attack instances. It reflects a common challenge in intrusion detection, where real attacks are rare compared to normal activity (Fig. 2). This imbalance can bias the classifier toward the BENIGN class, causing minority attacks to be overlooked during training. In practical intrusion detection scenarios, such a distribution is realistic because malicious traffic typically appears far less frequently than normal traffic. However, from a machine learning perspective, this condition may reduce recall and F1-Score for rare classes, particularly Brute Force, XSS, and SQL Injection. Therefore, imbalance handling becomes essential to ensure that the model remains sensitive to low-frequency but security-critical attacks.

TABLE I
SUMMARY OF THE CICIDS 2017 WEB ATTACK DATASET.

Records	Columns	Non-Null	Missing Values
458,968	85	170,366	288,602

B. Data Cleaning

The initial cleaning process involves removing one duplicate record from the BENIGN class, resulting in 170,365 unique records. A total of 20 missing values

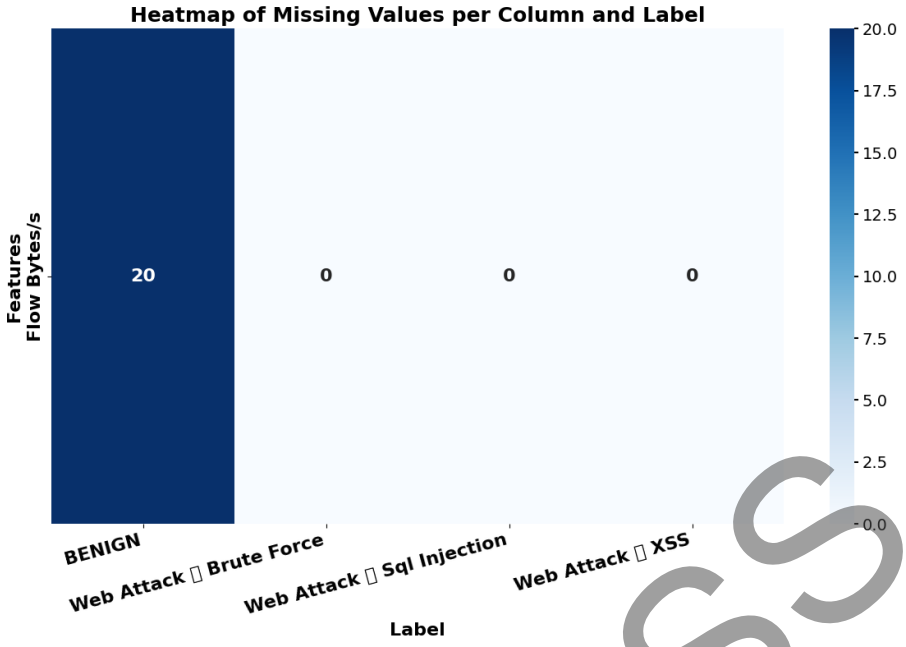

Fig. 3. Heatmap of missing values per column and label. Note: Structured Query Language (SQL) and Cross-Site Scripting (XSS).

is identified exclusively in the Flow Bytes/s feature, all within the BENIGN class. To prevent training bias caused by class-specific sparsity, this feature is eliminated (Fig. 3). It is consistent with prior studies, emphasizing the removal of features when missingness is confined to the majority class [25].

C. Removal of Irrelevant Feature

Attributes irrelevant to behavioral classification, such as flow ID, source/destination IP, timestamp, and port identifiers, are discarded due to their session-specific nature. These variables introduce noise and undermine generalizability by encoding non-informative, context-dependent patterns. This refinement reduces the number of features from 85 to 79 without semantic loss, thereby improving model interpretability and reducing the risk of overfitting.

D. Handling of Infinite Values

The analysis identifies 135 infinite values in Flow Bytes/s and Flow Packets/s, exclusively within BENIGN instances (Fig. 4). These anomalies likely stem from capture artifacts, such as session termination errors. All affected entries are removed to preserve distributional integrity and prevent majority-class learning bias, in line with prior recommendations [26].

E. Handling of Negative Values

Semantically invalid negative values are observed in several features, notably Init_Win_bytes_forward (81,911) and Init_Win_bytes_backward (102,355), predominantly within BENIGN traffic (Fig. 5). Since such values violate protocol semantics, targeted imputation is applied. The window size values are replaced with zero, while continuous variables are imputed using the median. This approach, validated in anomaly-handling literature [27], preserves statistical fidelity.

F. Protocol Field Filtering

Exploration of the protocol feature reveals three values: 0.0 (invalid), 6.0 (TCP), and 17.0 (UDP). Notably, the 0.0 value appears only in BENIGN-class records and is absent in attack classes (Table II), indicating to capture artifacts or reconstruction failures. Including such class-specific values risks data leakage by enabling the model to exploit non-semantic cues.

To preserve model generalizability, all records with protocol value 0.0 are removed, retaining only valid transport-layer protocols (Transmission Control Protocol (TCP) and User Datagram Protocol (UDP)). This step aligns with previous researchers who have excluded low-frequency categorical values to maintain dataset integrity and avoid overfitting due to spurious correlations [28].

G. Outlier Detection and Handling

To mitigate distortion from extreme values, outlier detection is conducted using the Interquartile Range (IQR) method. Outliers are defined as values that fall

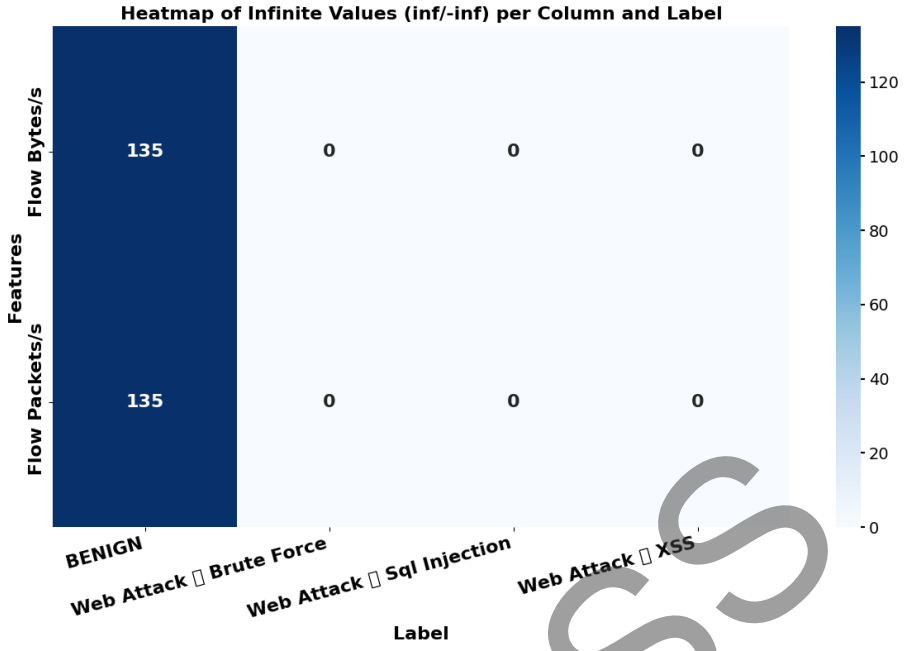

Fig. 4. Heatmap of Infinite Values (inf/-inf) per column and label. Note: Structured Query Language (SQL) and Cross-Site Scripting (XSS).

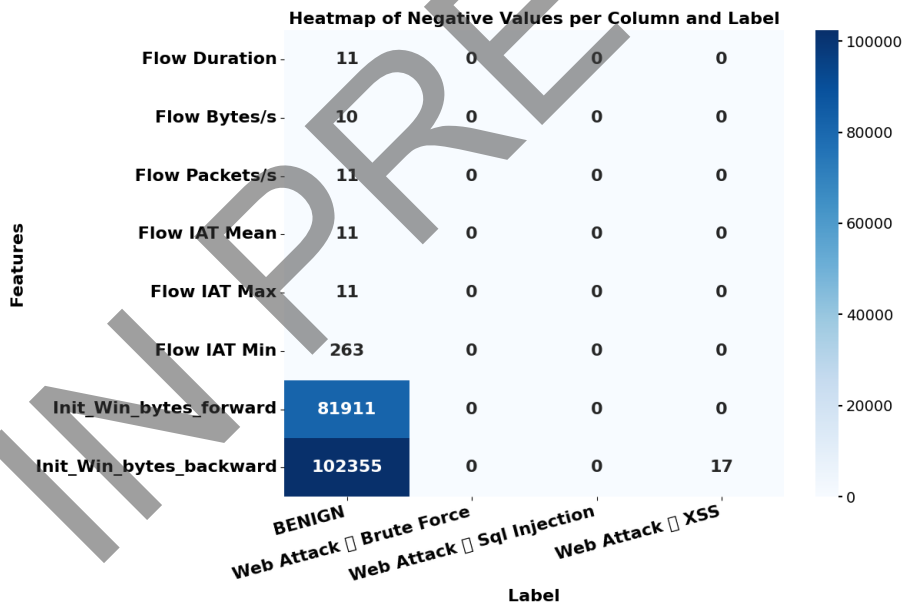

Fig. 5. Heatmap of negative values per column and label. Note: Inter-Arrival Time (IAT), Structured Query Language (SQL), and Cross-Site Scripting (XSS).

TABLE II
PROTOCOL VALUES BY TARGET LABEL IN CICIDS-2017 WEB ATTACK DATASET.

Protocol	BENIGN	Brute Force	Structured Query Language (SQL) Injection	Cross-Site Scripting (XSS)
0.0	141	0	0	0
6.0	86,142	1,507	21	652
17.0	81,767	0	0	0

below or above the thresholds. They are specified in Eqs. (1) and (2).

$$\text{Lower Bound} = Q_1 - 1.5 \times \text{IQR}, \tag{1}$$

$$\text{Upper Bound} = Q_3 + 1.5 \times \text{IQR}. \tag{2}$$

The IQR represents the difference between the third quartile (Q_3) and the first quartile (Q_1). These formulas are widely referenced in empirical research and constitute a standard data-trimming technique [29]. Several features, including Flow Duration, Flow Bytes/s, and Flow Inter-Arrival Time (IAT) Max, exhibited heavy-tailed distributions with extreme outliers, likely resulting from logging anomalies or rare network behaviors. Rather than removing these values, which can lead to information loss, a winsorization strategy is applied by capping values at the calculated boundary thresholds while preserving data volume and structure. This technique maintains the overall data distribution while reducing skewness and limiting the influence of noisy extremes, thereby enhancing model robustness and training stability.

H. Label Encoding

Prior to model training, the categorical target labels are encoded into integer format, It is to ensure compatibility with machine learning algorithms, which inherently operate on numerical inputs. This transformation preserves class semantics while enabling seamless integration into the classification pipeline.

I. Splitting and Normalization

The encoded dataset is partitioned into training (70%) and testing (30%) subsets using stratified sampling to preserve class distribution across the splits. This strategy enhances model stability, particularly for imbalanced multiclass datasets, as supported by previous research [30]. The choice of a 70:30 stratified split is further examined through preliminary experiments comparing alternative ratios (60:40, 70:30, and 80:20). The 70:30 configuration consistently achieves a favorable balance between training sufficiency and unbiased evaluation, with macro-F1 variations across ratios remaining within $\pm 0.5\%$. Given these results and its widespread adoption in IDS research, the 70:30 split is retained, as it provides stable generalization performance without overfitting.

Next, model generalizability is further assessed using 10-fold stratified cross-validation on the training data. It ensures robust performance estimation and minimizing sampling bias. This validation scheme is suitable for both balanced and imbalanced datasets [31].

To standardize feature magnitudes and ensure that all numerical attributes contribute proportionally during the modeling process, Min-Max scaling is applied to transform each feature into the [0, 1] range using Eq. (3) [32]. This normalization technique is important because the original features may have different scales, which can negatively affect the learning performance of machine learning algorithms. In Eq. (3), X' denotes the normalized value after scaling, while X represents the original value of the feature before transformation. Furthermore, $X_{\min}$ and $X_{\max}$ indicate the minimum and maximum values of the corresponding feature, respectively.

$$X' = \frac{X - X_{\min}}{X_{\max} - X_{\min}}. \tag{3}$$

J. Feature Selection

To optimize classification performance while reducing model complexity, feature selection is conducted using the ANOVA F-test. It evaluates the statistical separability of features across target classes. The F-statistic is represented in Eq. (4). The F denotes the F-statistic used to evaluate whether there are significant differences between group means. Meanwhile, MSB refers to the mean square between groups, which represents the variation among the group means, and MSW refers to the mean square within groups, which represents the variation within each group. A higher F value indicates that the variation between groups is greater than the variation within groups, suggesting that the feature has stronger discriminatory power.

$$F = \frac{MSB}{MSW}. \tag{4}$$

Following best practices in prior research [33], the top 20 features ranked by F-statistic are retained to balance predictive performance and computational efficiency (Fig. 6). Although the ANOVA F-test evaluates features individually, it does not capture inter-feature dependencies. Further multivariate analysis reveals overlapping distributions between the Brute Force and XSS attack classes. To address this limitation, density plots are examined to identify additional discriminative features.

Five attributes, Down/Up Ratio, Init_Win_bytes_forward, Backward (Bwd) Packets/s, Flow Bytes/s, and Flow IAT Min, demonstrate significant divergence in class-wise distributions (Fig. A1 in Appendix), confirming their value in enhancing class separability. This finding aligns with previous researchers [34]. They have emphasized the importance of domain-informed feature augmentation in imbalanced settings.

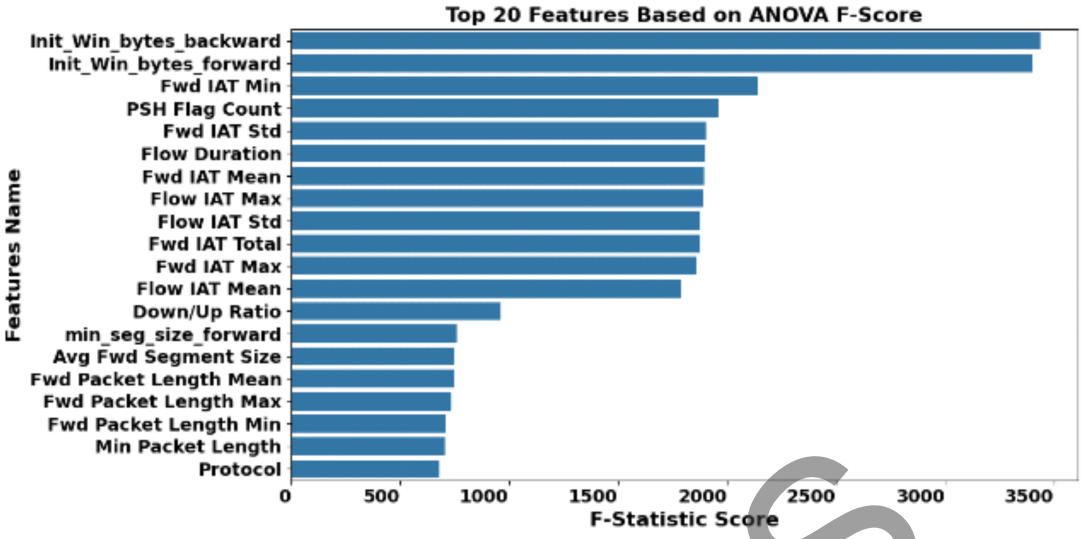

Fig. 6. Top 20 feature rankings based on Analysis of Variance (ANOVA) F-statistic value. Note: Inter-Arrival Time (IAT), Forward (Fwd), and Push Flag (PSH).

TABLE III
SUMMARY OF CLASS DISTRIBUTIONS BEFORE AND AFTER APPLYING SYNTHETIC MINORITY OVERSAMPLING TECHNIQUE (SMOTE).

Class	Before SMOTE	After SMOTE
BENIGN	117,635	117,635
Brute Force	1,055	117,635
Cross-Site Scripting (XSS)	456	117,635
Structured Query Language (SQL) Injection	15	117,635

Accordingly, these features are incorporated into the final model, expanding the feature set from 20 to 23. To determine the optimal weighting of these discriminative features, an algorithmic optimization procedure is applied. Then, a grid search is conducted over weight values ranging from 0.0 to 2.0 in increments of 0.1, using the macro-averaged F1-Score from 10-fold stratified cross-validation on the training set as the selection criterion.

K. Imbalanced Data

Severe class imbalance in the CICIDS-2017 Web Attack subset poses a challenge to effective model learning, particularly in detecting underrepresented threats such as Brute Force, XSS, and SQL Injection. To address this issue, the SMOTE is employed. Unlike random oversampling, SMOTE generates new synthetic samples by interpolating between a minority-class instance and one of its nearest neighbors, as defined in Eq. (5). This approach enhances the diversity of the minority-class distribution without replicating existing samples, thereby reducing the risk of overfitting and preserving the overall data manifold [35]. In Eq. (5), x_{new} denotes the newly generated synthetic sample, while x_i represents the selected minority class instance used as the reference point. The parameter λ is a random value in the interval $[0, 1]$ that determines the position of the synthetic sample along the line segment between two minority instances. Meanwhile, x_j denotes one of the nearest neighbors of x_i from the same minority class. This formulation enables the generation of synthetic data points to improve class balance in the dataset.

$$x_{\text{new}} = x_i + \lambda \cdot (x_j - x_i). \tag{5}$$

The application of SMOTE is proven highly effective. It is supported by previous research which have demonstrated substantial improvements in multiclass intrusion detection accuracy. SMOTE successfully balances the dataset by upsampling all minority classes to match the dominant BENIGN class [36]. Table III summarizes the class distributions before and after SMOTE application, with each class normalized to 117,635 samples.

L. Base Learner

The research employs a hybrid base-learner architecture by combining LightGBM and Random Forest, two ensemble-based algorithms that are complementary in terms of learning dynamics, scalability, and robustness. The integration of these models enhances detection capability in high-dimensional, imbalanced, and noisy network environments typical of intrusion

detection tasks. LightGBM, a gradient boosting framework, is optimized for speed and memory efficiency through histogram-based learning and leaf-wise tree growth. The model is updated iteratively by minimizing residual errors, as defined in Eq. (6). This iterative refinement enables adaptive error correction at each stage, facilitating convergence and performance improvement [37]. In Eq. (6), $F_m(x)$ denotes the updated prediction function at the m-th iteration, while $F_{m-1}(x)$ represents the prediction function obtained from the previous iteration. The parameter γ_m is the learning rate or step size that controls the contribution of the weak learner at iteration m. Meanwhile, $h_m(x)$ denotes the weak learner or base model fitted at the m-th stage to reduce the residual error from the previous model. This iterative formulation allows the model to gradually improve its predictive performance by adding corrective components in each boosting step.

$$F_m(x) = F_{m-1}(x) + \gamma_m \, h_m(x). \tag{6}$$

Complementarily, Random Forest aggregates the outputs of multiple uncorrelated Decision Trees. Each is trained on bootstrapped samples with randomly selected feature subsets. The final prediction $H(x)$ is obtained by averaging, as shown in Eq. (7):

$$H(x) = \frac{1}{2} \sum_{i=1}^{n} h_i(x), \tag{7}$$

Where $h_i(x)$ denotes the prediction of the i-th tree, and n represents the total number of trees. This aggregation mechanism contributes to low variance and high generalizability, which are particularly valuable for heterogeneous network traffic [38, 39].

The synergy between LightGBM's gradient-based precision and Random Forest's robustness to noise forms a resilient ensemble foundation, effectively capturing non-linear patterns and irregular class distributions inherent in web-based intrusion datasets. LightGBM is capable of identifying complex relationships among features through its boosting mechanism, enabling more precise classification of subtle attack patterns. In contrast, Random Forest contributes to the stability of the model by reducing overfitting and maintaining strong performance in the presence of noisy and high-dimensional data. By combining these complementary strengths, the ensemble model can achieve better generalization and improved detection capability across diverse intrusion scenarios.

M. Meta Learner

To integrate the outputs of the heterogeneous base learners, MLP is employed as the meta-learner, leveraging its capacity to model complex non-linear relationships among feature interactions. This architectural choice enhances classification performance, particularly in scenarios involving overlapping class boundaries and imbalanced distributions. The MLP is implemented as a feedforward neural network comprising fully connected layers. The input consists of the concatenated probabilistic outputs of the base models. Its forward propagation process is formally expressed in Eq. (8). The y denotes the final output of the neural network after passing through the hidden and output layers. The matrix W_1 represents the weight parameters connecting the input vector X to the hidden layer, while b_1 is the bias term added to the hidden layer transformation. Furthermore, W_2 denotes the weight parameters connecting the hidden layer to the output layer, and b_2 is the bias term associated with the output layer. The function $f(\cdot)$ represents the activation function applied to introduce non-linearity into the model, allowing the network to learn more complex patterns from the input data.

$$y = f(W_2 \cdot f(W_1 \cdot X + b_1) + b_2) \tag{8}$$

The selection of an MLP as the meta-learner is further supported by its demonstrated success in related domains, such as time-series forecasting and autoencoder-based systems [40], underscoring its suitability for capturing subtle distinctions in high-dimensional intrusion data. An MLP is particularly effective as a meta-learner because it can model complex non-linear relationships among the outputs generated by multiple base learners. This capability allows the model to integrate diverse prediction patterns and improve the final classification performance. In addition, the hidden-layer architecture of an MLP enables it to learn more abstract feature representations, which is beneficial for distinguishing between normal and malicious traffic patterns. Therefore, the use of an MLP in the stacking framework provides greater flexibility and enhances the overall robustness of the intrusion detection model.

N. Hyperparameter Optimization

To ensure optimal predictive performance and generalization of the MLP meta-learner, Bayesian hyperparameter optimization is conducted using the TPE. The TPE algorithm models the conditional probability $p(x \mid y)$ of a hyperparameter configuration x, given a performance score y, as shown in Eq. (9) [41]. In Eq. (9), $l(x)$ denotes the density function over hyperparameter configurations x associated with objective values better than the predefined threshold y^*. Meanwhile, $g(x)$ represents the density function over hyperparameter configurations corresponding to objec-

TABLE IV
HYPERPARAMETER SEARCH SPACE USED IN TREE-STRUCTURED PARZEN ESTIMATORS (TPE) OPTIMIZATION.

Parameter	Value Type	Range/Choices	Description
Hidden layer sizes	Discrete choice	(1–50), (1–50)	Number and size of hidden layers
Activation	Discrete choice	'relu', 'tanh', 'logistic'	Activation function used in the Multi-Layer Perceptron (MLP)
Solver	Discrete choice	'adam', 'sgd','lbfgs'	Optimization algorithm used for training
Alpha	Continuous	Uniform between 0.001 and 0.00001	L2 regularization strength
Learning rate init	Continuous	Uniform between 0.001 and 0.00001	Initial learning rate

tive values worse than or equal to y^*. The symbol y^* refers to the threshold value of the objective function used to divide the observations into promising and less promising regions during the TPE optimization process. By modeling $l(x)$ and $g(x)$ separately, TPE can more efficiently identify hyperparameter configurations that are likely to improve the performance of the MLP meta-learner.

$$p(x \mid y) = \begin{cases} l(x), & \text{if } y < y^*, \\ g(x), & \text{if } y \geq y^*. \end{cases} \tag{9}$$

The search space is carefully defined to include the key MLP parameters, as outlined in Table IV. In the research, the LightGBM and Random Forest classifiers are used with their default hyperparameters as implemented in the LightGBM and Scikit-learn libraries. For LightGBM, the parameters are set as follows: num_leaves = 31, max_depth = -1, learning_rate = 0.1, n_estimators = 100, min_child_samples = 20, subsample = 1.0, colsample_bytree = 1.0, reg_alpha = 0.0, and reg_lambda = 0.0. For Random Forest, the parameters are: n_estimators = 100, criterion = 'gini', max_depth = None, min_samples_split = 2, min_samples_leaf = 1, max_features = 'sqrt', and bootstrap = True. SMOTE was applied with k_neighbors = 5, sampling_strategy = 1.0, and random_state = 42. The same random seed is used throughout preprocessing and training to ensure reproducibility. No additional hyperparameter tuning is performed for the base learners, as the primary objective is to evaluate the fundamental effectiveness of the stacking framework. Preliminary experiments indicates competitive performance under the default configurations.

O. Evaluation

Macro-averaged metrics are adopted to mitigate class imbalance by assigning equal weight to all classes, including minority attacks (SQL Injection and XSS). Performance is estimated via 10-fold stratified cross-validation and finalized on a held-out test set to ensure a robust, unbiased assessment. The evaluation spanned multiclass (BENIGN, Brute Force, XSS, SQL Injection) and binary (BENIGN vs. ATTACK) settings to examine scalability and granularity. Results are reported using confusion matrices, performance curves, and SHAP-derived feature importance to identify key contributors. The memory footprint is quantified with `psutil` on Ubuntu 22.04, capturing model size on disk and Resident Set Size (RSS) at baseline, after data loading, after model loading, and during inference under the same environment and batch configuration as the latency study.

III. RESULTS AND DISCUSSION

A. Results

The proposed stacked ensemble model outperforms the individual base learners (Random Forest and LightGBM) across all key evaluation metrics. As shown in Fig. 7, it achieves a macro-averaged precision of 0.70, recall of 0.80, and an F1-score of 0.62, compared to 0.54 (Random Forest) and 0.50 (LightGBM). This result highlights its superior generalization, particularly under class imbalance.

The performance gain is further amplified through TPE-based hyperparameter optimization. The optimal MLP meta-learner configuration (Rectified Linear Unit (ReLU) activation, $\alpha = 0.0001048$, hidden layers $= (5, 25)$, learning rate ≈ 0.0009677, solver $=$ `adam`) significantly boosts sensitivity to minority classes. This improvement indicates that the selected hyperparameter combination enables the MLP to learn more representative decision boundaries for underrepresented attack categories. The use of the TPE approach is beneficial because it efficiently explores the search space and identifies promising hyperparameter settings without requiring exhaustive evaluation of all possible combinations. As a result, the optimized meta-learner achieves better generalization performance and contributes to a more balanced intrusion detection capability across different class distributions.

Despite an overall accuracy of $\geq 98\%$ across all models, macro-F1-Score is prioritized due to its robustness in imbalanced settings. However, confusion matrix analysis (Fig. 8) reveals persistent misclassification between minority classes. It is shown notably in Brute Force and XSS, attributed to overlap in feature distributions.

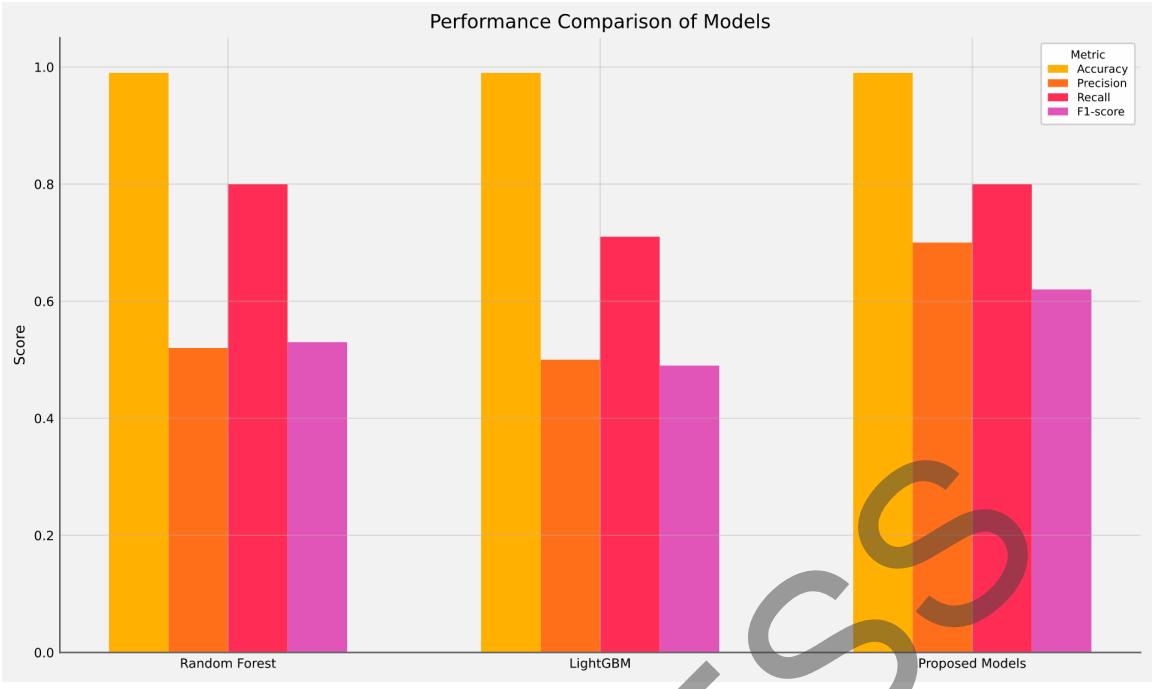

Fig. 7. Comparison of macro-averaging performances of the base learners and the stacking hybrid model.

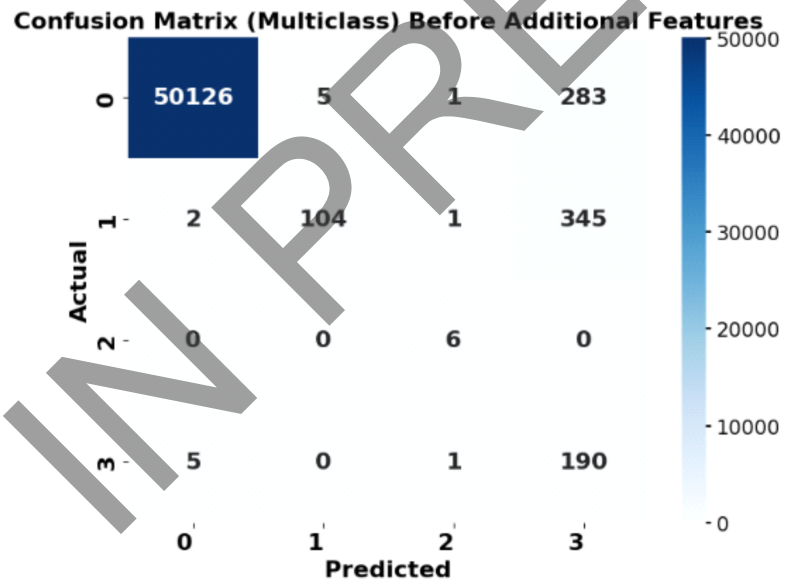

Fig. 8. Confusion matrix before adding more features. Misclassification of Brute Force as Cross-Site Scripting (XSS) highlights substantial overlap in feature distributions between these classes.

Applying the optimally derived weight to the five additional discriminative features (Fig. 9) significantly improves minority-class detection. The macro F1-Score peaks at a weight of 0.1 (0.77), outperforming all other tested weights in the 0.0–2.0 range. Under this setting, Brute Force (label 1) achieves a recall of 0.71 and an F1-Score of 0.70. Then, SQL Injection (label 2) reaches perfect recall (1.00) with an F1-Score of 0.92. Meanwhile, XSS (label 3) improves to a recall of 0.56 and an F1-Score of 0.46. The BENIGN (label 0) maintains perfect accuracy. Overall, this configuration delivers a macro recall of 0.82, a macro F1-Score of 0.77, and 99% accuracy, confirming that targeted reweighting of selected features enhances

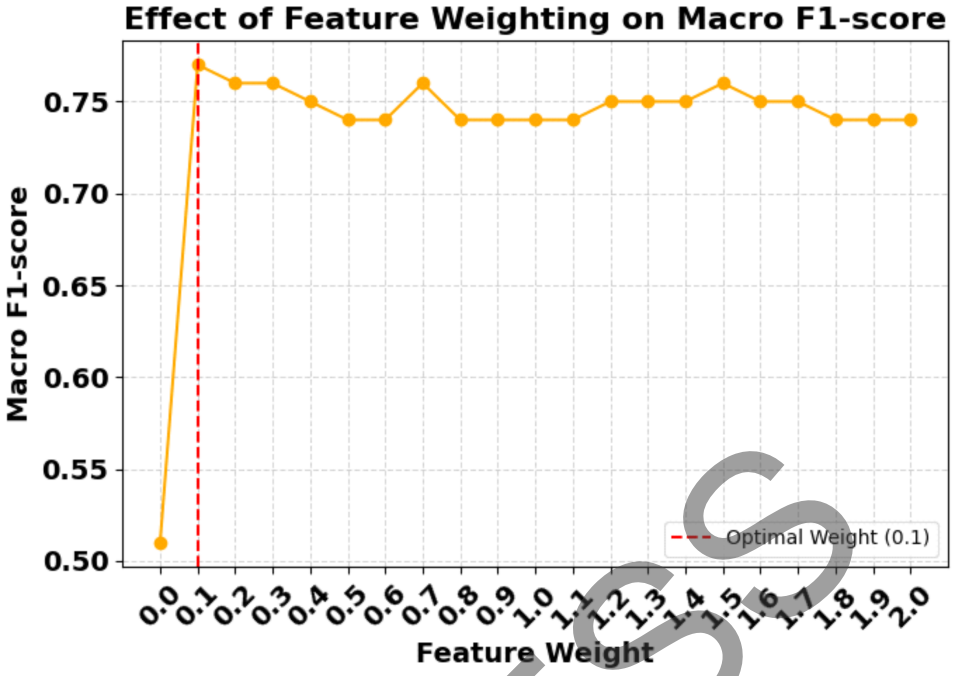

Fig. 9. Effect of varying the weight (0.0–2.0) applied to the five additional discriminative features on macro-averaged F1-Score.

minority-class detection without degrading majority-class performance.

Figures 10 and 11 and Table V further validate the impact of feature augmentation and the optimal weighting value, showing improved class balance and model sensitivity across all attack types. The results indicate that feature augmentation helps to enrich the representation of minority-class patterns, allowing the model to better distinguish less frequent attack categories. In addition, the application of the optimal weighting value reduces the bias toward majority classes and improves the classifier's responsiveness to minority instances. These findings demonstrate that the combination of augmented features and appropriate weighting contributes to a more balanced and reliable intrusion detection performance.

TABLE V
PER-CLASS METRICS BEFORE AND AFTER FEATURE AUGMENTATION.

Desc.	Metric	Before	After
BENIGN	Precision	1.00	1.00
	Recall	0.99	1.00
	F1-Score	1.00	1.00
Brute Force	Precision	0.95	0.68
	Recall	0.23	0.71
	F1-Score	0.37	0.70
SQL Injection	Precision	0.67	0.86
	Recall	1.00	1.00
	F1-Score	0.80	0.92
XSS	Precision	0.23	0.38
	Recall	0.97	0.56
	F1-Score	0.37	0.46
Macro Avg.	Precision	0.71	0.73
	Recall	0.80	0.82
	F1-Score	0.64	0.77
Weighted Avg.	Precision	1.00	0.99
	Recall	0.99	0.99
	F1-Score	0.99	0.99

Note: Structured Query Language (SQL) and Cross-Site Scripting (XSS).

To evaluate broader generalization, the model is also assessed in a binary setting (BENIGN vs. ATTACK). In post-enhancement, it achieves 99.72% accuracy, 0.9888 macro recall, and 0.9498 macro F1-Score, with 641 of 654 attack instances correctly classified (Fig. 12 and Table VI). This result demonstrates the model's adaptability across different classification settings. The high macro recall indicates that the model is able to identify attack instances effectively, including those that are relatively limited in number. In addition, the strong macro F1-Score shows that the model maintains a good balance between precision and recall after enhancement. These findings confirm that the proposed approach not only preserves classification precision but also becomes more responsive to minority threats in a simplified binary detection scenario.

Collectively, these findings affirm the effectiveness of the stacked hybrid framework, particularly the role of feature engineering and reweighting in addressing class imbalance. Feature engineering contributes by enhancing the representational quality of the input data,

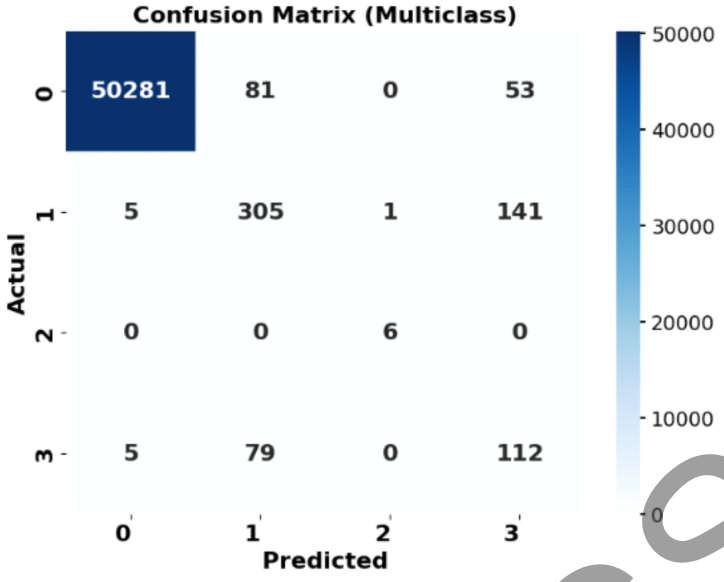

Fig. 10. Confusion matrix following the addition of the discriminative features and the weight adjustment. Both Brute Force and Cross-Site Scripting (XSS) attacks show improved detection.

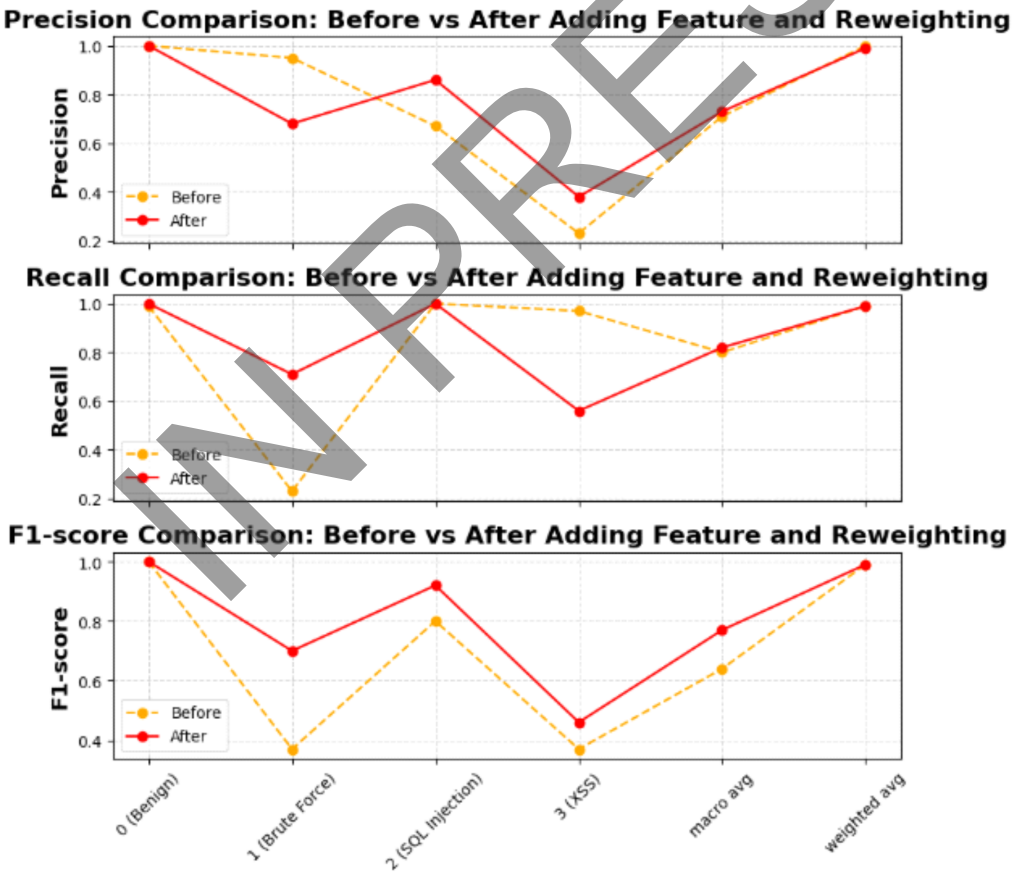

Fig. 11. Comprehensive comparison of pre-feature and post-feature augmentation. Note: Structured Query Language (SQL) and Cross-Site Scripting (XSS).

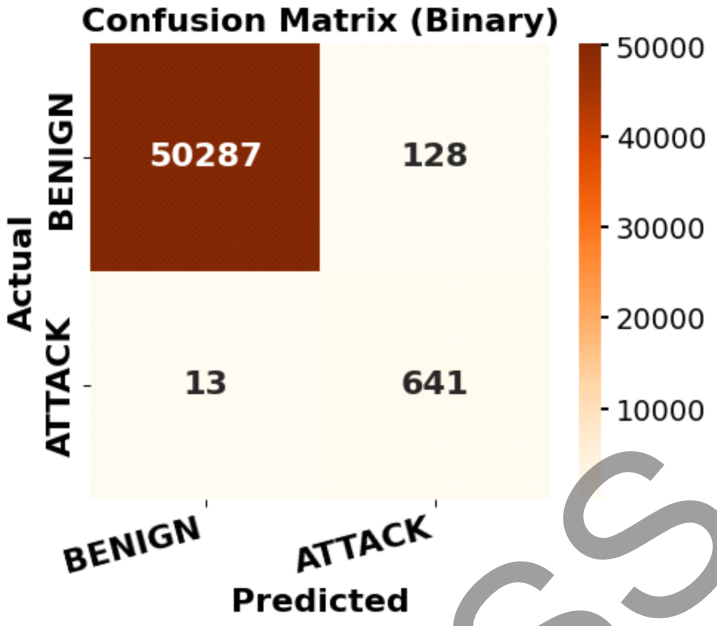

Fig. 12. Binary classification confusion matrix, indicating enhanced recognition of ATTACK instances following augmentation of the features.

TABLE VI
PERFORMANCE BINARY CLASSIFICATION SCENARIO.

Class	Precision	Recall	F1-score
BENIGN (0)	0.9997	0.9975	0.9986
ATTACK (1)	0.8336	0.9801	0.9009
Macro Avg.	0.9166	0.9888	0.9498
Weighted Avg.	0.9972	0.9972	0.9972

allowing the model to better capture discriminative patterns across both majority and minority classes. Meanwhile, the reweighting strategy helps to reduce the dominance of majority-class instances during training, thereby improving the model's sensitivity to less frequent attack types. These complementary mechanisms strengthen the overall robustness of the framework and support its potential for real-time IDS deployment in highly skewed environments.

To verify that the performance improvement of the proposed stacking model over the strongest baseline is not due to random chance, a statistical significance test is conducted. Among the baseline models evaluated in the research, Random Forest achieves the highest macro-averaged F1-Score on the held-out test set. Therefore, it is selected as the strongest baseline for comparison. A 5-fold stratified cross-validation on the training set produces macro-averaged F1-Scores of 0.9260 ± 0.00046 for the proposed stacking model and 0.9164 ± 0.00076 for the Random Forest baseline. The fold-wise scores are compared using a paired Student's t-test, yielding a t-statistic of 24.88 and a p-value of 1.55×10^{-5}. These results indicate that the observed improvement is statistically significant and unlikely to have occurred by chance.

To ensure model interpretability, SHAP (SHapley Additive exPlanations) analysis is applied to the MLP meta-learner, revealing that decisions are predominantly influenced by Flow IAT Min, Init_Win_bytes_forward, and Init_Win_bytes_backward across all classes. For Brute Force, packet symmetry (Bwd Packets/s) and window size patterns are key factors, while XSS is driven by flow timing metrics such as Down/Up Ratio and Flow Bytes/s. SQL Injection exhibits minimal SHAP contributions, reflecting its relative separability (Figs. 13–16).

Table VII lists the top 10 features by average SHAP value per class. It confirms that the model leverages attack-specific behavioral patterns rather than spurious correlations. This layer of explainability not only reinforces the model's validity but also enhances its operational trustworthiness for real-world deployment in security-critical systems.

B. Discussion

The results affirm the efficacy of the stacked ensemble architecture, LightGBM and Random Forest as base learners, with an MLP as the meta-learner, in addressing class imbalance and improving minority

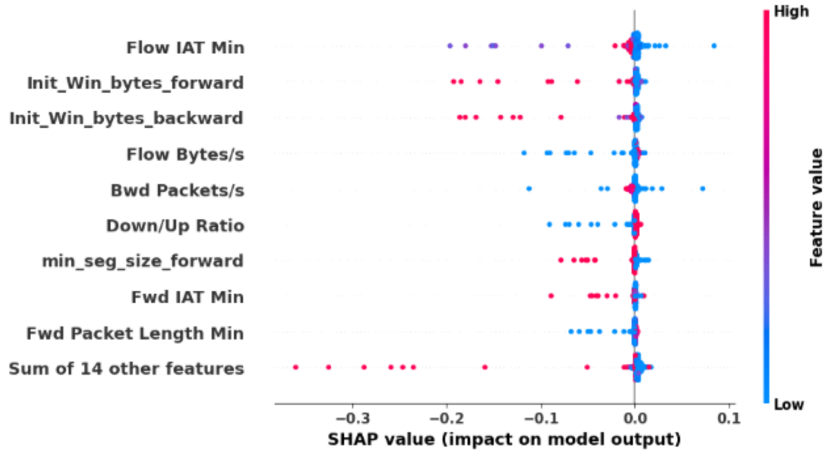

Fig. 13. SHapley Additive exPlanations (SHAP) summary plot for class 0 (BENIGN). Note: Inter-Arrival Time (IAT), Backward (Bwd), and Forward (Fwd).

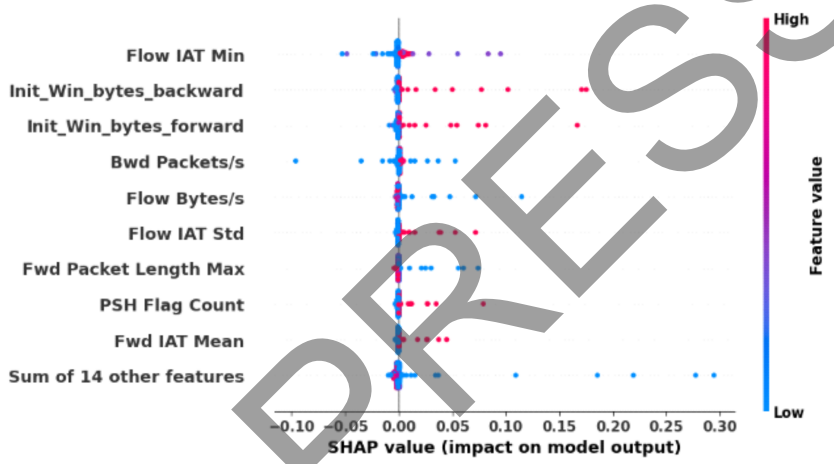

Fig. 14. SHapley Additive exPlanations (SHAP) summary plot for class 1 (Brute Force). Note: Inter-Arrival Time (IAT), Backward (Bwd), and Forward (Fwd).

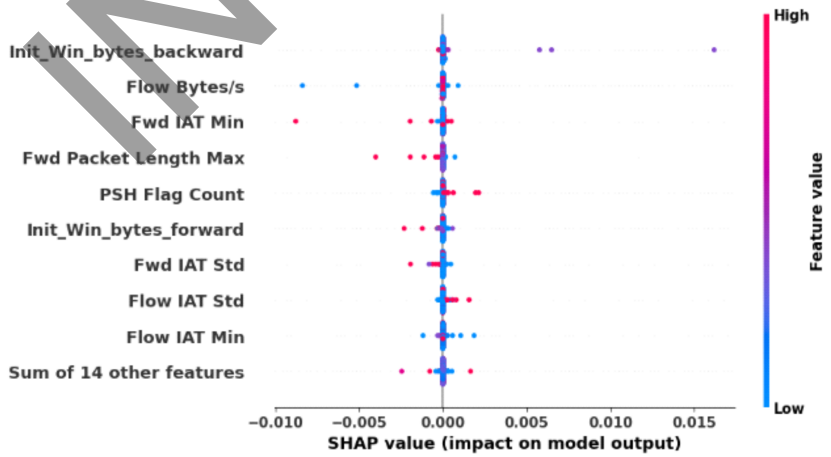

Fig. 15. SHapley Additive exPlanations (SHAP) summary plot for class 2 (SQL Injection). Note: Inter-Arrival Time (IAT), Backward (Bwd), and Forward (Fwd).

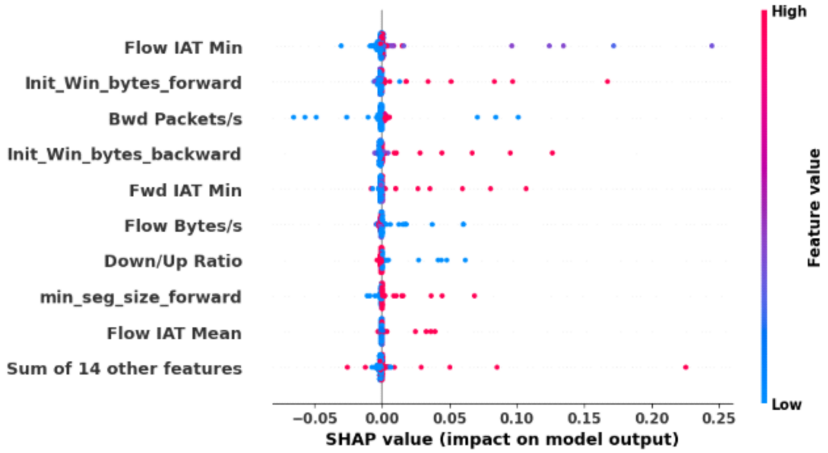

Fig. 16. SHapley Additive exPlanations (SHAP) summary plot for class 3 (Cross-Site Scripting (XSS)). Note: Inter-Arrival Time (IAT), Backward (Bwd), and Forward (Fwd).

TABLE VII
TOP 10 FEATURES BY AVERAGE SHAPLEY ADDITIVE EXPLANATIONS (SHAP) VALUE PER CLASS.

Rank	Feature	Class 0	Class 1	Class 2	Class 3
1	Flow IAT Min	0.005272	0.002855	0.000016	0.002890
2	Init_Win_bytes_forward	0.003892	0.001728	0.000019	0.002189
3	Init_Win_bytes_backward	0.003842	0.002075	0.000062	0.001758
4	Flow Bytes/s	0.002458	0.001271	0.000034	0.001194
5	Bwd Packets/s	0.002219	0.001379	–	0.002101
6	Down/Up Ratio	0.001778	–	–	0.001024
7	min_seg_size_forward	0.001629	–	–	0.000897
8	Fwd IAT Min	0.001223	–	0.000028	0.001404
9	Fwd Packet Length Min	0.001198	0.000761	–	–
10	Fwd IAT Mean	0.001072	0.000817	–	0.000609

Note: Inter-Arrival Time (IAT), Backward (Bwd), and Forward (Fwd).

class detection in the CICIDS-2017 dataset. Achieving a macro F1-score of 0.76, the proposed model substantially outperforms standalone classifiers (Random Forest: 0.53; LightGBM: 0.26), validating the strength of model integration (Fig. 7).

Notably, the Brute Force and SQL Injection classes exhibit significant gains: recall increases from 0.23 to 0.71 and the F1-Score from 0.37 to 0.70 for Brute Force, while SQL Injection achieves perfect recall (1.00) with an improved F1-score (0.92). Although XSS recall declines from 0.97 to 0.56, the gain in precision from 0.23 to 0.38 indicates improved true positive discrimination. It highlights a favorable shift in the precision–recall trade-off.

Persistent misclassification between Brute Force and XSS is traced to overlapping feature distributions, particularly in Flow Bytes/s, Flow IAT Min, and Bwd Packets/s (Fig. A1 in Appendix). Their targeted inclusion and weighted amplification ($\times 0.1$) demonstrably improves minority class separation (Figs. 9 and 11). The result underscores the impact of feature-level adjustments in enhancing model sensitivity.

The performance gains achieved by the proposed stacking model are primarily attributed to its strategic integration of three core components: optimized preprocessing, a hybrid ensemble architecture, and targeted feature engineering. Compared to prior models in Table VIII, such as Op-ReDMAT and EFedID, which rely heavily on single deep architectures or federated learning, this framework combines LightGBM and Random Forest as base learners, balancing speed, generalization, and robustness. Meanwhile, the MLP meta-learner enhances decision boundaries through nonlinear aggregation. The superior macro F1-Score of 99.35% in multiclass classification confirms the effectiveness of this heterogeneous stacking scheme.

Furthermore, the model benefits from extensive preprocessing steps, including outlier mitigation, normalization, and ANOVA-based feature selection, which stabilize training and reduce noise. Unlike prior models, the researchers incorporate domain-informed feature augmentation and class-specific reweighting. It proves critical in improving minority class detection, especially for Brute Force and XSS attacks, which

TABLE VIII
COMPARISON OF EVALUATION RESULTS WITH PREVIOUS STUDIES IN MULTICLASS SCENARIO.

Models	Dataset	Multi Class			
		Accuracy	Precision	Recall	F1-Score
Op-ReDMAT [13]	CICIDS 2017 (all)	99.12%	98.6%	98.2%	98.8%
EFedID [42]	CICIDS 2017 (all)	95.51%	96.5%	96%	96.2%
Ensemble (FA-CNN + DAE) [9]	CICIDS 2017 (all)	95%	–	–	–
Semisupervised (AC + K-Means + Voting) [14]	CICIDS 2017 (DDoS)	96.66%	97%	–	–
K-Nearest Neighbors (KNN) [15]	CICIDS 2017 (web attack)	96%	96%	96%	96%
Naïve Bayes [15]	CICIDS 2017 (web attack)	96%	96%	96%	96%
Decision Tree [15]	CICIDS 2017 (web attack)	96%	96%	96%	96%
Random Forest [15]	CICIDS 2017 (web attack)	98%	98%	98%	98%
AdaBoost [15]	CICIDS 2017 (web attack)	98%	98%	98%	98%
Proposed Models	CICIDS 2017 (web attack)	99.28%	99.43%	99.29%	99.35%

Note: Optimized Resource-Efficient Deep Mutual Attention Transformer (Op-ReDMAT), Ensemble Federated Intrusion Detection (EFedID), Feature Augmentation (FA), Convolutional Neural Network (CNN), Denoising Autoencoder (DAE), Agglomerative Clustering (AC), and Distributed Denial of Service (DDoS).

TABLE IX
COMPARISON OF EVALUATION RESULTS WITH PREVIOUS STUDIES IN BINARY SCENARIO.

Models	Dataset	Binary Class			
		Accuracy	Precision	Recall	F1-Score
DMAE (DAE + Magnet Loss) + RF classifiers [16]	CICIDS 2017 (web attack)	97.8%	96.1%	96.1%	96.1%
Proposed Models	CICIDS 2017 (web attack)	99.727%	99.72%	99.72%	99.72%

Note: Deep Marginal Autoencoder (DMAE) and Random Forest (RF).

TABLE X
PERFORMANCE OF THE PROPOSED MODEL ACROSS MULTIPLE DATASETS.

Dataset	Attack Type(s)	Evaluation			
		Accuracy	Precision	Recall	F1-Score
CICIDS-2017 DDoS	DDoS	99%	99%	99%	99%
CICIDS-2017 Infiltration	Infiltration	99%	99%	99%	99%
UNSW-NB15	Various	100%	100%	100%	100%
MTA-KDD '19	Various	100%	100%	100%	100%
KDD Cup'99	Various	96%	99%	90%	94%

are typically underrepresented. This focus on class separability and balanced representation leads to substantial gains over traditional classifiers, such as KNN, Naïve Bayes, and Random Forest or AdaBoost, which plateaue at 96–98% accuracy. In binary classification (Table IX), the model further outperforms complex deep architectures, such as Deep Marginal Autoencoder (DMAE) + Random Forest, achieving a 99.69% F1-Score while maintaining low latency, highlighting its suitability for real-time applications.

It is also benchmarked on other widely recognized IDS datasets such as UNSW-NB15 and KDD Cup '99, which are frequently used in ensemble IDS studies, to evaluate the model's generalizability. These datasets have been extensively analyzed in prior literature. For example, previous research [43] has provided a comprehensive evaluation of UNSW-NB15 for network intrusion detection, while another research [44] has analyzed the KDD dataset. As summarized in Table X, the model achieves high accuracy and weights F1-Score across most datasets. It reaches perfect scores (1.00) on CICIDS-2017 DDoS, UNSW-NB15, and MTA-KDD '19. Strong results are also obtained on CICIDS-2018 (0.90), KDD Cup '99 (0.96), and CICIDS-2017 Infiltration (1.00). Meanwhile, NSL-KDD yields a lower weighted F1-Score of 0.76. It is expected due to its outdated and imbalanced structure. These results confirm the model's robustness and adaptability across diverse traffic types and attack patterns.

A runtime evaluation confirms computational efficiency and deployment feasibility. Tested on a VPS running Ubuntu Server 22.04 with a 6-core Intel Xeon E5-1650 @ 3.20 GHz and 8 GB RAM, the system achieves a total inference time of 0.6652 s across the test set, averaging 0.000013 s per sample and a throughput of 76,923.0769 samples per second. Using `psutil`, the memory footprint is quantified as follows: serialized model size of 586.24 MB; Resident Set Size (RSS) baseline of 137.20 MB; RSS after data loading of 273.65 MB; RSS after model loading

of 1.02 GB; and an RSS peak during inference of 1.07 GB, with a steady-state usage of 1.07 GB. These results fall within the 8 GB RAM budget and support real-time IDS deployment on resource-constrained servers.

In essence, the research achieves strong classification accuracy while also demonstrating significant potential for the development of effective and adaptive cybersecurity models which are capable of identifying complex web-based attacks in dynamic network environments. This result indicates that the proposed framework is not only reliable in terms of predictive performance, but also flexible enough to handle variations in traffic patterns and attack characteristics. The combination of ensemble learning, feature engineering, and data balancing strategies strengthens the model's capability to detect both common and less frequent intrusion types. Therefore, the findings support the applicability of the proposed approach as a robust foundation for intelligent intrusion detection systems in evolving cybersecurity contexts.

IV. Conclusion

The research introduces a stacking-based intrusion detection framework combining LightGBM, Random Forest, and MLP meta-learner, optimized via TPE and reinforced with a rigorous data preprocessing and feature engineering pipeline. Through the integration of ANOVA-based feature selection, contextual feature augmentation, and class-specific weighting, the framework significantly improves recognition of minority attacks such as Brute Force and XSS. The model demonstrates robust generalization across multiple benchmark datasets while maintaining sub-millisecond inference time, thereby showing strong suitability for real-time deployment in resource-limited environments. Notably, the proposed model improves the macro F1-Score from 0.62 to 0.77 in multiclass classification and achieves 99.72% accuracy with sub-millisecond latency, significantly enhancing minority attack detection (e.g., Brute Force recall improves from 0.23 to 0.71).

The research limitations include the reliance on default configurations for the base learners and SMOTE parameters. IIt may limit the model's ability to achieve the best possible performance, as different hyperparameter settings can further improve classification results and class balance. In addition, the use of a single meta-learning architecture may restrict the exploration of other advanced models that can better capture complex relationships in intrusion data. Therefore, more comprehensive hyperparameter optimization and the investigation of alternative meta-learning architectures, such as TabNet or transformer-based models, remain important directions for future research.

In future research, the current framework can be extended by incorporating alternative meta-learning architectures such as TabNet or transformer-based models, which are capable of modeling more intricate feature interactions. These architectures may provide greater representational power and improve the model's ability to capture complex dependencies among intrusion-related features. Additionally, while the present research employs default configurations for LightGBM, Random Forest, SMOTE, and random seed settings to establish a controlled baseline, future research should investigate the impact of comprehensive hyperparameter tuning for these components. Such exploration is important to identify more optimal parameter combinations that could further enhance classification accuracy, robustness, and minority-class detection performance.

Acknowledgement

The authors would like to express their sincere gratitude to the Canadian Institute for Cybersecurity (CIC), University of New Brunswick, for providing access to the CICIDS-2017 dataset used in this research. The authors also thank Diponegoro University for its academic support and research environment that facilitated the completion of this study. This research was conducted independently without specific funding from public, commercial, or not-for-profit agencies.

Author Contribution

Developed the stacking framework (RF + LightGBM with an MLP meta-learner) and designed the experimental protocol on CICIDS-2017 web attacks, F. D. P.; Collected and curated the CICIDS-2017 web attack subset and prepared training/testing splits, F. D. P.; Implemented preprocessing, SMOTE, and TPE-based hyperparameter optimization and built the evaluation pipeline, F. D. P.; Conducted model training, ablation, statistical testing, and runtime/memory profiling and interpreted the results, F. D. P.; Wrote and revised the manuscript and prepared all figures and tables, F. D. P.; Provided guidance on the study design and experimental protocol, including recommendations on model selection, evaluation metrics, and validation strategy, F.; Reviewed and validated the experimental results and contributed to the interpretation of the findings, particularly the comparative evaluation against baseline models, F.; Reviewed and edited the manuscript and improved the scientific writing, presentation of results, and overall coherence, F.; Supervised the research process and provided academic oversight throughout the study, F.; Proposed validity testing and methodological

design, including recommendations for stratified cross-validation, statistical significance testing, and evaluation using macro-averaged F1-Score, B. W.; Contributed to the analysis by validating the experimental results and providing critical interpretation of the findings, particularly regarding model generalization and minority-class performance, B. W.; Reviewed the manuscript for clarity and technical correctness, B. W.; and Provided academic guidance and oversight, with emphasis on methodological rigor and validity testing, B. W.

Data Availability

The CICIDS-2017 dataset used in the research is publicly available from the Canadian Institute for Cybersecurity (CIC) at the University of New Brunswick: https://www.unb.ca/cic/datasets/ids-2017.html. The experiments in the research are conducted on the CICIDS-2017 Web Attack subset. The processed dataset (after cleaning, scaling, feature selection, and SMOTE), trained model artifacts, and evaluation outputs generated are available from the corresponding author upon reasonable request.

References

[1] H. Hindy, R. Atkinson, C. Tachtatzis, J.-N. Colin, E. Bayne, and X. Bellekens, "Utilising deep learning techniques for effective zero-day attack detection," *Electronics*, vol. 9, no. 10, pp. 1–16, 2020.

[2] N. Chergui and N. Boustia, "Contextual-based approach to reduce false positives," *IET Information Security*, vol. 14, no. 1, pp. 89–98, 2020.

[3] Z. K. Maseer, R. Yusof, N. Bahaman, S. A. Mostafa, and C. F. M. Foozy, "Benchmarking of machine learning for anomaly based intrusion detection systems in the CICIDS2017 dataset," *IEEE Access*, vol. 9, pp. 22 351–22 370, 2021.

[4] A. Choubey and A. V. N. Krishna, "Intrusion detection system using deep learning methodologies," *Journal of Mathematical and Computational Science*, vol. 11, no. 5, pp. 5278–5295, 2021.

[5] Y. C. Wang, Y. C. Houng, H. X. Chen, and S. M. Tseng, "Network anomaly intrusion detection based on deep learning approach," *Sensors*, vol. 23, no. 4, pp. 1–21, 2023.

[6] D. K. Kang, K. B. Lee, and Y. C. Kim, "Cost efficient GPU cluster management for training and inference of deep learning," *Energies*, vol. 15, no. 2, pp. 1–20, 2022.

[7] A. Ito, K. Saito, R. Ueno, and N. Homma, "Imbalanced data problems in deep learning-based side-channel attacks: Analysis and solution," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 3790–3802, 2021.

[8] S. Basodi, C. Ji, H. Zhang, and Y. Pan, "Gradient amplification: An efficient way to train deep neural networks," *Big Data Mining and Analytics*, vol. 3, no. 3, pp. 196–207, 2020.

[9] B. Selvakumar, M. Sivaanandh, K. Muneeswaran, and B. Lakshmanan, "Ensemble of feature augmented convolutional neural network and deep autoencoder for efficient detection of network attacks," *Scientific Reports*, vol. 15, pp. 1–17, 2025.

[10] K. V. K. Chithanya and L. Reddy, "Automatic intrusion detection model with secure data storage on cloud using adaptive cyclic shift transposition with enhanced ANFIS classifier," *Cyber Security and Applications*, vol. 3, pp. 1–15, 2025.

[11] A. S. Dina and D. Manivannan, "Intrusion detection based on machine learning techniques in computer networks," *Internet of Things*, vol. 16, 2021.

[12] S. M. Kasongo, "A deep learning technique for intrusion detection system using a recurrent neural networks based framework," *Computer Communications*, vol. 199, pp. 113–125, 2023.

[13] M. H. Alsulami, "Residual dense optimization-based multi-attention transformer to detect network intrusion against cyber attacks," *Applied Sciences*, vol. 14, no. 17, p. 7763, 2024.

[14] M. Aamir and S. M. A. Zaidi, "Clustering based semi-supervised machine learning for DDoS attack classification," *Journal of King Saud University-Computer and Information Sciences*, vol. 33, no. 4, pp. 436–446, 2021.

[15] M. H. Almourish, O. A. Abduljalil, and A. E. B. Alawi, "Anomaly-based web attacks detection using machine learning," in *International Conference on Smart Computing and Cyber Security: Strategic Foresight, Security Challenges and Innovation*. Wonju-si, South Korea: Springer, Oct. 28–29, 2021, pp. 306–314.

[16] V. Q. Nguyen, V. H. Nguyen, T. C. Nguyen, and N. Shone, "A novel deep learning approach with magnet loss optimization for website attack detection," in *2024 1st International Conference on Cryptography and Information Security (VCRIS)*. Hanoi, Vietnam: IEEE, Dec. 3–4, 2024, pp. 1–6.

[17] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[18] ——, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[19] D. H. Wolpert, "Stacked generalization," *Neural*

Networks, vol. 5, no. 2, pp. 241–259, 1992.

[20] E. Sevgen and S. Abdikan, "Classification of large-scale mobile laser scanning data in urban area with LightGBM," *Remote Sensing*, vol. 15, no. 15, pp. 1–19, 2023.

[21] H. Wang, "Research on the application of random forest-based feature selection algorithm in data mining experiments," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 10, pp. 505–518, 2023.

[22] M. Massaoudi, S. S. Refaat, I. Chihi, M. Trabelsi, F. S. Oueslati, and H. Abu-Rub, "A novel stacked generalization ensemble-based hybrid LGBM-XGB-MLP model for short-term load forecasting," *Energy*, vol. 214, 2021.

[23] K. Vamsi Krishna, K. Swathi, P. Rama Koteswara Rao, and B. Basaveswara Rao, "A detailed analysis of the CIDDS-001 and CICIDS-2017 datasets," in *Pervasive Computing and Social Networking: Proceedings of ICPCSN 2021*. Salem, India: Springer, 2022, pp. 619–638.

[24] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP 2018)*. Madeira, Portugal: SCITEPRESS – Science and Technology Publications, Lda., Jan. 22–24, 2018, pp. 108–116.

[25] Z. Wang, C. F. Tsai, and W. C. Lin, "Data cleaning issues in class imbalanced datasets: Instance selection and missing values imputation for one-class classifiers," *Data Technologies and Applications*, vol. 55, no. 5, pp. 771–787, 2021.

[26] Z. Ning, Z. Jiang, and D. Zhang, "Sparse projection infinite selection ensemble for imbalanced classification," *Knowledge-Based Systems*, vol. 262, 2023.

[27] G. Kabir, S. Tesfamariam, J. Hemsing, and R. Sadiq, "Handling incomplete and missing data in water network database using imputation methods," *Sustainable and Resilient Infrastructure*, vol. 5, no. 6, pp. 365–377, 2020.

[28] H. Chamlal, T. Ouaderhman, and F. Aaboub, "A graph based preordonnances theoretic supervised feature selection in high dimensional data," *Knowledge-Based Systems*, vol. 257, 2022.

[29] L. K. Mramba, X. Liu, K. F. Lynch, J. Yang, C. A. Aronsson, S. Hummel, J. M. Norris, S. M. Virtanen, L. Hakola, U. M. Uusitalo, and J. P. Krischer, "Detecting potential outliers in longitudinal data with time-dependent covariates," *European Journal of Clinical Nutrition*, vol. 78, no. 4, pp. 344–350, 2024.

[30] A. Rácz, D. Bajusz, and K. Héberger, "Effect of dataset size and train/test split ratios in QSAR/QSPR multiclass classification," *Molecules*, vol. 26, no. 4, pp. 1–16, 2021.

[31] T. Fontanari, T. C. Fróes, and M. Recamonde-Mendoza, "Cross-validation strategies for balanced and imbalanced datasets," in *Brazilian Conference on Intelligent Systems*. Campinas, Brazil: Springer, Nov. 28–Dec. 1, 2022, pp. 626–640.

[32] M. A. Siddiqi and W. Pak, "An agile approach to identify single and hybrid normalization for enhancing machine learning-based network intrusion detection," *IEEE Access*, vol. 9, pp. 137 494–137 513, 2021.

[33] S. S. Panwar, Y. P. Raiwani, and L. S. Panwar, "An intrusion detection model for CICIDS-2017 dataset using machine learning algorithms," in *2022 International Conference on Advances in Computing, Communication and Materials (ICACCM)*. Dehradun, India: IEEE, Nov. 10–11, 2022, pp. 1–10.

[34] V. Madhumithaa and J. Govindarajan, "Domain based network intrusion detection system for IoT," in *2023 IEEE 7th Conference on Information and Communication Technology (CICT)*. IEEE, 2023, pp. 1–7.

[35] F. Kamalov, S. E. Choutri, and A. F. Atiya, "Analytical formulation of Synthetic Minority Oversampling Technique (SMOTE) for imbalanced learning," *Gulf Journal of Mathematics*, vol. 19, no. 1, pp. 400–415, 2025.

[36] A. O. Widodo, B. Setiawan, and R. Indraswari, "Machine learning-based intrusion detection on multi-class imbalanced dataset using SMOTE," *Procedia Computer Science*, vol. 234, pp. 578–583, 2024.

[37] I. K. Nti, O. Narko-Boateng, A. F. Adekoya, and A. R. Somanathan, "Stacknet based decision fusion classifier for network intrusion detection." *The International Arab Journal of Information Technology*, vol. 19, no. 3A, pp. 478–490, 2022.

[38] E. M. Hameed, H. Joshi, and A. A. A. Ismael, "The effect of combining datasets in diabetes prediction using ensemble learning techniques," *CommIT (Communication and Information Technology) Journal*, vol. 19, no. 1, pp. 129–140, 2025.

[39] H. Pham and S. Olafsson, "On cesaro averages for weighted trees in the random forest," *Journal of Classification*, vol. 37, no. 1, pp. 223–236, 2020.

[40] X. Xiao, J. Liu, D. Liu, Y. Tang, J. Dai, and F. Zhang, "SSAE-MLP: Stacked sparse autoencoders-based multi-layer perceptron for main bearing temperature prediction of large-scale wind turbines," *Concurrency and Computation: Practice and Experience*, vol. 33, no. 17, 2021.

[41] M. Liang, B. An, K. Li, L. Du, T. Deng, S. Cao, Y. Du, L. Xu, X. Gao, L. Zhang, J. Li, and H. Gao, "Improving genomic prediction with machine learning incorporating TPE for hyperparameters optimization," *Biology*, vol. 11, no. 11, pp. 1–13, 2022.

[42] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *2015 Military Communications and Information Systems Conference (MilCIS)*. Canberra, ACT, Australia: IEEE, Nov. 10–12, 2015, pp. 1–6.

[43] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*. Ottawa, ON, Canada: IEEE, July 8–10, 2009, pp. 1–6.

[44] N. He, Z. Zhang, X. Wang, and T. Gao, "Efficient privacy-preserving federated deep learning for network intrusion of industrial IoT," *International Journal of Intelligent Systems*, vol. 2023, no. 1, pp. 1–22, 2023.

APPENDIX

The Appendix can be seen in the next page.

Fig. A1. Distribution of additional feature density between Brute Force and Cross-Site Scripting (XSS) classes. Note: Inter-Arrival Time (IAT) and Backward (Bwd).