

RO-DAD: a resource-optimized DDoS attack detection framework using multi-stage feature selection in cloud computing

Questa è la versione sottoposta a revisione paritaria (postprint) della seguente opera:

Original

RO-DAD: a resource-optimized DDoS attack detection framework using multi-stage feature selection in cloud computing / Haque Mansoor, U.I., Durad Muhammad, H., Usman, A., Khan Farrukh, A., Afzal, S.. - In: JOURNAL OF CLOUD COMPUTING. - ISSN 2192-113X. - (2026). [10.1186/s13677-026-00959-6]

Availability:

This version is available at: 20.500.11771/44698

Publisher:

Springer Nature

Published

DOI:10.1186/s13677-026-00959-6

Terms of use:

This publication is made accessible in accordance with the terms for deposit in the institutional repository, as defined by the IMT School for Advanced Studies Lucca's Open Access Policy.
(https://library.imtlucca.it/sites/default/files/regolamento-policy-open-access-imtlib_0.pdf).

Si prega di consultare le pagine informative dell'editore relative alle politiche di autoarchiviazione.

(Article begins on next page)

RO-DAD: A resource-optimized DDoS attack detection framework using multi-stage feature selection in cloud computing

Received: 20 December 2025

Accepted: 16 July 2026

Published online: 11 August 2026

Cite this article as: Haque M.u., Durad M.H., Usman A. *et al.* RO-DAD: A resource-optimized DDoS attack detection framework using multi-stage feature selection in cloud computing. *J Cloud Comp* (2026). <https://doi.org/10.1186/s13677-026-00959-6>

Mansoor ul Haque, Muhammad Hanif Durad, Anila Usman, Farrukh Aslam Khan & Sara Afzal

We are providing an unedited version of this manuscript to give early access to its findings. Before final publication, the manuscript will undergo further editing. Please note there may be errors present which affect the content, and all legal disclaimers apply.

If this paper is publishing under a Transparent Peer Review model then Peer Review reports will publish with the final article.

RO-DAD: A Resource-Optimized DDoS Attack Detection Framework Using Multi-Stage Feature Selection in Cloud Computing

Mansoor-ul-haque^{1,2}, Muhammad Hanif Durad^{1,2}, Anila Usman¹, Farrukh Aslam Khan³, and Sara Afzal^{4,5,*}

¹Department of Computer and Information Sciences, Pakistan Institute of Engineering and Applied Sciences (PIEAS), Islamabad 45650, Pakistan

²Critical Infrastructure Protection and Malware Analysis (CIPMA) Lab, Pakistan Institute of Engineering and Applied Sciences (PIEAS), Islamabad 45650, Pakistan

³Center of Excellence in Information Assurance, King Saud University, Riyadh 11653, Saudi Arabia

⁴Institute of Informatics and Telematics (IIT) - National Research Council (CNR), Pisa, Italy

⁵IMT School for Advanced Studies, Lucca, Italy

*sara.afzal@iit.cnr.it

ABSTRACT

Cloud computing has transformed the modern IT infrastructure with its scalable, cost-effective, and on-demand services. Distributed Denial-of-Service (DDoS) attacks, however, are growing in number and sophistication, and they can adversely affect service availability, negatively impact performance, cause financial losses, and damage reputation, particularly in the cloud environment. Conventional detection methods are not always adequate to deal with the dynamic and massive nature of these attacks. To solve these challenges, this paper proposes a machine learning-based Resource-Optimized DDoS Attack Detection (RO-DAD) framework that enhances detection efficiency through a multi-stage feature selection pipeline. The proposed framework integrates Domain Knowledge (DK) filtering, Least Significant (LS) feature elimination, and Most Significant (MS) selection, where the MS stage is implemented using an embedded tree-based feature selection mechanism. This design enables systematic identification of the most informative features while reducing computational complexity and eliminating redundant attributes. The framework is assessed using the CICDDoS2019 dataset, and experimental findings show that the proposed approach achieves high detection performance, reaching up to 99% accuracy and F1-score, while significantly reducing the dimensionality of features and computational overhead. This confirms that effective feature optimization can maintain strong detection capability while enabling scalable and resource-efficient deployment in cloud environments.

Introduction

Cloud computing has greatly broadened the attack surface of today's digital infrastructures and thus has made cloud environments more and more attractive targets for cyber threats. Although substantial progress has been made in strengthening cloud security mechanisms like authentication, access control, and data confidentiality^{1,2}, maintaining service availability remains a critical concern. Among the threats affecting cloud-hosted services, Distributed Denial-of-Service (DDoS) attacks continue to pose a major challenge by overwhelming targeted resources with excessive traffic, thereby disrupting legitimate user access³. In addition to causing service outages, these attacks can also lead to financial losses, reduced quality of service, and reputation damage. The identification and mitigation of DDoS attacks are especially challenging because malicious traffic can have many similar traits to legitimate network traffic, making it difficult to accurately identify and respond to attacks in time. Modern DDoS attacks are massive and complex in nature, making it difficult to effectively deal with these attacks using rule-based and signature-based detection techniques in dynamic cloud environments^{4,5}. Therefore, the field of DDoS detection has become an area of great interest for using machine learning (ML) approaches that can process enormous amounts of network data and discover patterns associated with attacks based on historical data^{6,7}.

Unlike static detection mechanisms that rely on predefined rules, ML-based approaches can learn discriminative qualities from observed traffic and adapt to changing attack patterns. Recent studies have showed that various ML and deep learning (DL) algorithms can achieve good detection performance across diverse DDoS assault scenarios⁸⁻¹¹. Despite these advances, achieving high detection accuracy is insufficient for practical cloud deployment, where computational efficiency and resource utilization are equally important considerations.

Feature selection is crucial for enhancing the performance of ML-based DDoS detection systems in high-dimensional

network traffic data. By removing irrelevant and redundant attributes, it reduces computational overhead, improves model efficiency, and enables faster detection, which is essential for large-scale and cost-sensitive cloud environments. Despite these advantages, many existing DDoS detection studies continue to rely on high-dimensional feature spaces, which increases training complexity, memory consumption, and inference latency. Therefore, identifying compact and highly informative feature subsets remains an important challenge for developing scalable and Resource-Optimized DDoS detection frameworks¹². Moreover, many existing approaches rely on single-stage feature selection strategies that may still retain redundant or weakly informative attributes, limiting overall effectiveness.

In order to overcome these drawbacks, this study presents a multi-stage feature selection pipeline in order to reduce the feature dimensionality progressively while retaining the discriminative information, called Resource Optimized DDoS Attack Detection (RO-DAD) framework. The framework integrates Domain Knowledge (DK) filtering, Least Significant (LS) feature elimination, and Most Significant (MS) selection, enabling systematic removal of irrelevant and low-importance attributes. In this pipeline, an embedded feature selection mechanism is used based on tree-based learning models because they can capture non-linear relationships and can naturally obtain feature importance in the training process. Decision Trees (DT), Random Forests (RF), and XGBoost estimate feature relevance based on their contribution to impurity reduction, enabling effective identification of high-impact attributes in high-dimensional DDoS datasets^{13,14}. This allows the framework to prioritize informative features while reducing computational overhead, resulting in a compact feature subset that remains effective for detection tasks. The proposed framework is tested with the CICDDoS2019 dataset that includes a variety of modern and diverse DDoS attack scenarios and also offers a wide range of flow-based traffic features to test the detection models¹⁵. The key contributions of this paper are summarized as follows:

- We introduce a Resource-Optimized DDoS Attack Detection (RO-DAD) framework that includes multi-stage feature selection process using Domain Knowledge (DK), Least Significant (LS), and Most Significant (MS) filtering to reduce the feature dimensionality without compromising detection capability.
- We validate the proposed framework on the CICDDoS2019 dataset using multiple ML classifiers under diverse DDoS attack scenarios.
- We conduct detailed analysis of the trade-off between detection performance and computational efficacy with precision, recall, F1-score, accuracy, statistical significance testing, and feature reduction analysis.

The remainder of the paper is organized as follows: In Section II, related work on DDoS detection in cloud environment is reviewed. The proposed methodology, RO-DAD is presented in Section III. The dataset and experimental evaluation are described in Section IV, and the conclusion of the paper and future work in Section V.

Related Work

In cloud computing, users are able to use the resources remotely through a network. These resources are supposed to be available instantly and continuously¹⁶. This availability can be impacted by flooding the network or the resources themselves, and by making them unavailable when they are needed by legitimate users, during a DDoS attack¹⁷. Some of the latest and the most commonly used mechanisms in the literature for detecting DDoS assaults in cloud computing systems are briefly discussed in this section.

Rajagopal et al.¹⁸ investigated the deployment of ML-based network intrusion detection on the Azure cloud platform. Although their work showcases the practical implementation of intrusion detection systems (IDS), their reliance on Azure-specific tools may hinder their portability to other cloud environments. Thorat et al.¹⁹ have suggested a divide-and-conquer approach to classify DDoS attacks based on a structured taxonomy. The model can be used to enhance the accuracy of classification, but the complexity may hinder real-time detection due to higher computational requirements. Ali et al.²⁰ compared the performance of ML and DL models in recognizing DDoS attacks in Software Defined Networks. The study was successful in showing the advantages of both approaches, but there was a lack of emphasis on methodologies which could have led to increased detection rates. Alzahrani and Alenazi²¹ investigated the application of different ML algorithms in the analysis and identification of DDoS attacks. While it does provide a good overview of various types of ML, the analysis in the paper does not account for more recent developments that may enhance detection in distributed systems.

In their paper, Virupakshar et al.²² investigated DDoS attack detection, focusing on bandwidth and connection flooding. The authors suggest a system comprising OpenStack, a firewall, and raw socket monitoring. KNN, NB, DT, and DNN algorithms were compared using a controlled DDoS attack dataset, along with LAN and KDD Cup datasets, to evaluate detection performance. However, the feature set used in this paper to detect the attack can be extended to get better results. Gumaste et al.²³ present a distributed processing framework that uses ML classifiers for real-time DDoS attack detection. The authors test their approach on a cloud testbed using OpenStack. Their findings indicate that the RF method outperforms other

ML classifiers with respect to accuracy. However, it is crucial to recognize that the authors did not compare more advanced ML models, which are currently more prevalent and often yield superior outcomes in addressing such challenges. In the work by Dhanapal and Nithyanandam²⁴, a new approach to identifying DDoS attacks was introduced. The proposed solution is integrated into the OpenStack IaaS software to show its effectiveness. However, the authors did not compare their solution with any ML-based algorithm to show its effectiveness against the most recent ever-changing and sophisticated DDoS attacks. In a separate study, Haque et al.²⁵ used a hybrid ANN-AE model to detect DDoS assaults in cloud environments, using OpenStack to implement the testbed. However, the effectiveness of such models heavily depends on efficient feature selection, as incorporating irrelevant features can lead to increased computational overhead and reduced detection accuracy. Moreover, the study does not compare its results with publicly available benchmark datasets, such as CICDDoS2019, which limits the validation and generalizability of its findings.

Kushwah and Ranga²⁶ proposed a hybrid ML approach combining extreme learning machine (ELM) and adaptive differential evolution (ADE) for efficient DDoS attack detection in cloud environments. The model optimizes hidden layer weights and biases with ADE and analytically calculates the output layer weights, which increases the accuracy of detection. To provide a comprehensive evaluation, three benchmark datasets, NSL-KDD, ISCX IDS 2012 and CIDDs-001, were employed. The lack of more recent data such as CICDDoS2019, however, hinders its usefulness when it comes to current DDoS attack trends. Furthermore, the complexity of the model and its performance in real-world large-scale cloud environments may make it difficult to implement. In another study²⁷, the authors proposed a solution for detecting DDoS attacks using voting-based extreme learning machines (ELMs). However, the evaluation relied on an outdated dataset that does not represent contemporary cloud computing networks. Furthermore, the proposed technique was not compared with established ensemble algorithms.

Alqarni²⁸ proposed an ensemble approach to detect DDoS attacks by voting. This ensemble system employs base classifiers including NB, DT, SVM, and K-NN. The study aimed to specifically identify DDoS attacks in cloud computing environments. The results demonstrated that the ensemble strategy achieved higher accuracy than individual classifiers, particularly with smaller datasets and limited feature sets. Further evaluation using diverse dataset partitions and feature configurations is necessary to validate these findings. In another study, Mugunthan²⁹ combined Rényi entropy, a Hidden-Markov Model, and a RF classifier to develop a novel approach to network intrusion detection. The attack probabilities were calculated using Rényi entropy, the attack severity was predicted using the Hidden Markov Model and the network traffic was classified by the RF classifier with bootstrap aggregation. The model was tested on the KDD CUP 99 dataset, and the results showed that the model outperforms the ABC-ANN model and ATBA model in terms of accuracy. The data set KDD CUP 99, however, is old, and the results may not be applicable to today's network environments, including cloud computing.

Bahashwan et al.³⁰ performed a systematic literature review (SLR) of several DDoS attack detection and mitigation tactics. Their analysis was based on the ML, DL, and hybrid methods between 2014 and 2022. The review revealed that ML techniques were most popular, followed by DL and hybrid approaches, and ensemble-based methods for both ML and DL showed promising results. Aljuhani et al.³¹ discuss the dynamic issues of DDoS attacks and underline the need for effective mitigation techniques. Their study provides a comprehensive review of the current DDoS mitigation techniques, particularly focusing on ML and DL methods. Classifications of DDoS attacks, different networking environments such as cloud computing and the challenges of developing realistic DDoS defense systems in these environments are discussed.

As shown in the reviewed studies, these models have tremendous potential, but the real-time and cloud-based application of these models will rely on the ability to overcome the challenge of resource consumption by using the appropriate feature selection. Tree-based feature selection methods have been found to be particularly successful in ranking the importance of the features, and they enable the models to select the most relevant features while reducing the feature space. This leads to faster processing times, lower computational load and no compromise on performance. To improve the scalability and adaptability of feature selection, further research is needed to improve the techniques, especially in cloud environments where network traffic is expected to rise in complexity. By selecting the relevant features, the DDoS detection system can operate efficiently, ensuring that it does not consume excessive computational resources or become overwhelmed by the sheer volume of data.

Methodology

Preliminaries

Machine Learning (ML)

ML is a subfield of AI that allows systems to learn patterns from data and make predictions without being explicitly programmed. In the field of network security, ML methods have demonstrated excellent performance in modeling intricate network traffic patterns, making them suitable for intrusion detection and DDoS detection applications^{32,33}.

To test the robustness and generalization capability of the proposed RO-DAD framework, we test eight supervised ML algorithms: DT, RF, AdaBoost, XGBoost, Logistic Regression (LR), Gaussian Naive Bayes (GNB), Support Vector Machine (SVM), and a Stacked Ensemble. The models were selected to represent the learning paradigms and performance characteristics of a broad range of models. Non-linear models like DT, RF, AdaBoost and XGBoost are good choices for non-linear patterns

and class imbalance issues^{34–36}. LR has a simple and interpretable linear baseline, and GNB is computationally efficient in high dimensional spaces when independence assumptions hold true. SVM has been proven to perform well on generalization by using margin maximization³⁷ and the Stacked Ensemble improves the prediction performance by stacking multiple base learners⁷. This variety of models provides a thorough assessment of the proposed framework within various learning paradigms and levels of complexity.

Datasets

The dataset used for this study is mainly the CICDDoS2019 dataset. A brief comparison with several popular datasets, however, is provided below to explain the main similarities and differences with the CICDDoS2019 dataset, for comprehensive contextualization and comparative analysis.

NSL-KDD: This is an improved version of the KDD Cup 1999 dataset, which aims to overcome some of the limitations of the original data set. However, it is still not very expressive of the complexity and variety of modern network traffic.

UNSW-NB15: This dataset contains a variety of network traffic data, including normal traffic and different types of attacks. Although more recent than KDD Cup 1999, it may not entirely represent the complexities of contemporary DDoS attacks.

ISCX-IDS-2012: This dataset contains network traffic data for intrusion detection research, including DDoS attacks. It provides a more modern and diverse set of attack scenarios compared to older datasets.

CICIDS 2017: Designed primarily for intrusion detection systems, it comprises diverse network traffic data, encompassing both normal traffic and various attack types, including DDoS, brute force, and infiltration. It provides a detailed and realistic representation of network traffic, making it suitable for developing and evaluating IDS models.

CICDDoS2019: The dataset is specifically tailored for research on DDoS attacks. It includes a comprehensive collection of features related to DDoS attacks that covers a wide range of attack scenarios and network traffic patterns. This dataset is designed to help researchers develop and test DDoS detection techniques.

Our research focuses primarily on DDoS attacks, utilizing the CICDDoS2019 dataset as our core resource due to its scale, diversity, and realistic simulation of network traffic. The CICDDoS2019 dataset contains modern volumetric and protocol-based DDoS attack scenarios across multiple services and protocols, making it suitable for evaluating detection frameworks in realistic conditions.

The CICDDoS2019 dataset is organized into two distinct sessions: a testing day and a training day. The testing session contains approximately 20 million network flow instances across 7 DDoS attack types and benign traffic. The training session comprises over 50 million instances, covering 12 DDoS attack types along with benign traffic. A detailed breakdown of all traffic classes is provided in Table 1.

Performance Metrics

The proposed framework is assessed using accuracy, precision, recall, F1-score, weighted F1-score, and the confusion matrix^{38–40}. TP, TN, FP, and FN denote correctly and incorrectly classified attack and normal instances. Accuracy, precision, recall, and F1-score are computed using Equations 1–4. To evaluate computational efficiency, training time, prediction time, and peak memory consumption are measured using Equations 5, 6, and 7, respectively.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

$$precision = \frac{TP}{TP + FP} \quad (2)$$

$$recall = \frac{TP}{TP + FN} \quad (3)$$

$$F1 - Score = \frac{2 * precision * recall}{precision + recall} \quad (4)$$

$$\Delta t_{train} = t_{t_end} - t_{t_start} \quad (5)$$

Table 1. Attacks Distribution of CICDDoS2019 Dataset

Sets	Class Name	Count / Instances
Testing	MSSQL	5787453
	Syn	4891500
	UDP	3867155
	NetBIOS	3657497
	LDAP	1915122
	Portmap	186960
	BENIGN	56965
	UDPLag	1873
Training	TFTP	20082580
	DrDoS_SNMP	5159870
	DrDoS_DNS	5071011
	DrDoS_MSSQL	4522492
	DrDoS_NetBIOS	4093279
	DrDoS_UDP	3134645
	DrDoS_SSDP	2610611
	DrDoS_LDAP	2179930
	Syn	1582289
	DrDoS_NTP	1202642
	UDP-lag	366461
	BENIGN	56863
	WebDDoS	439

$$\Delta t_{pred} = t_{p_end} - t_{p_start} \quad (6)$$

$$m_{train} = m_{peak} - m_{baseline} \quad (7)$$

To improve readability, the symbols and variables used in the computational efficiency metrics are summarized in Table 2.

Table 2. List of Notations and their Explanation

Notation	Explanation
Δt_{train}	Model training time.
Δt_{pred}	Model prediction time.
t_{t_start}	Training start time.
t_{t_end}	Training end time.
t_{p_start}	Prediction/testing start time.
t_{p_end}	Prediction/testing end time.
m_{train}	Memory consumed during model training.
m_{peak}	Peak memory usage observed during training.
$m_{baseline}$	Initial memory usage before training starts.

Proposed RO-DAD Framework Architecture

The Resource-Optimized DDoS Attack Detection (RO-DAD) framework is a systematic architecture designed to enhance the detection of Distributed Denial of Service (DDoS) attacks while optimizing computational resource usage. As illustrated in Figure 1, the RO-DAD framework follows a modular pipeline that includes dataset preparation, data pre-processing, class balancing, multi-stage feature filtering, and final model training and evaluation.

Each stage is carefully designed to reduce noise, eliminate irrelevant or redundant features, and enhance model interpretability and performance. Special emphasis is placed on feature reduction through domain knowledge, statistical analysis, and model-driven ranking. This approach not only enhances detection accuracy but also lowers computational costs. This structured pipeline guarantees that the model effectively detects DDoS attacks while optimizing resource use.

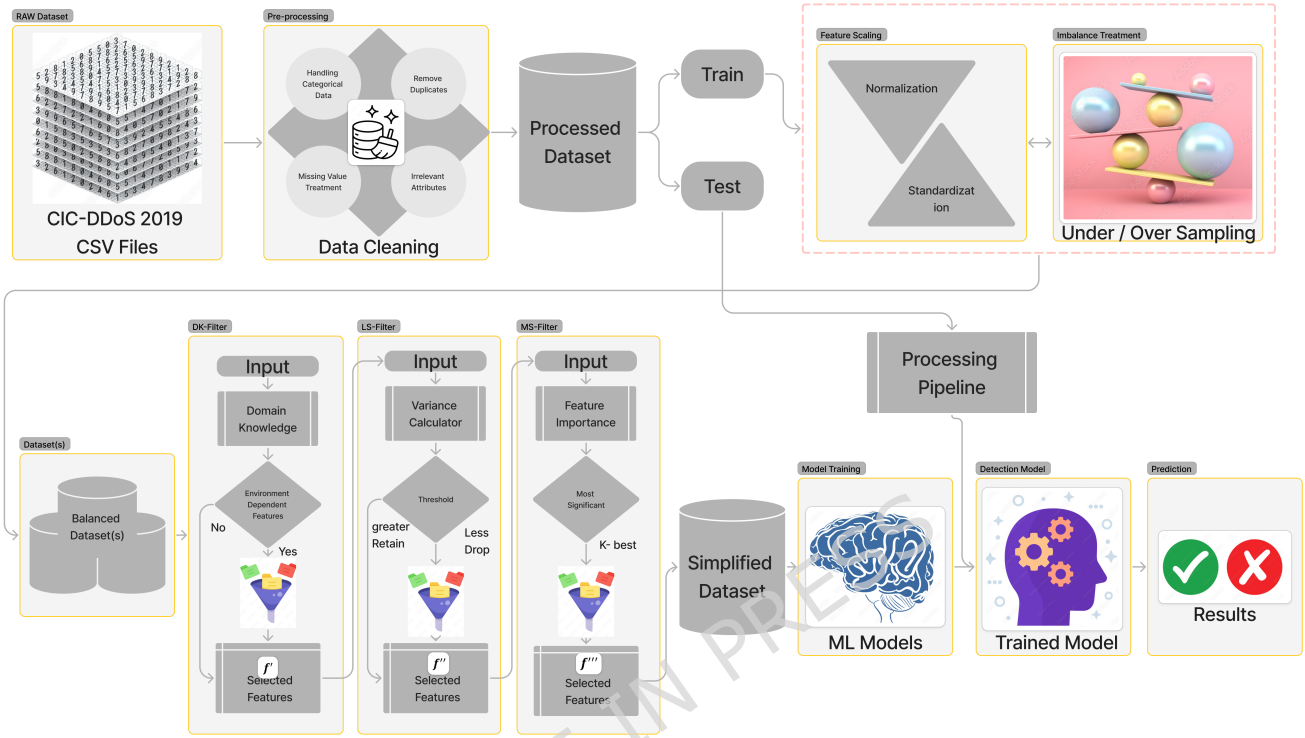


Figure 1. RO-DAD Framework Architecture: A multi-stage pipeline for efficient and accurate DDoS attack detection, incorporating domain knowledge, statistical filtering, and tree-based feature ranking.

Pre-processing The CICDDoS2019 dataset was first cleaned and prepared for analysis by removing irrelevant records and ensuring consistency across all feature attributes. This study utilizes the testing-day portion of the CICDDoS2019 dataset because it contains all attack categories considered in this work and provides a sufficiently large number of instances for controlled experimentation. Furthermore, using a single traffic subset minimizes variability introduced by traffic collected on different days, allowing the impact of the proposed balancing and feature selection procedures to be evaluated more consistently.

Rather than adopting the original dataset partitioning, a new stratified train-test split (70/30) was performed to maintain class distribution across all attack categories and facilitate a fair comparison of the proposed framework under different balancing strategies. The resulting training and testing sets were mutually exclusive, ensuring that no testing samples were used during model training. The objective of this study is to investigate feature optimization and computational efficiency in DDoS detection rather than cross-day traffic generalization.

The objective of this study is to investigate feature optimization and computational efficiency under controlled conditions rather than cross-day traffic generalization. Consequently, a stratified train-test split was performed on a single traffic subset to isolate the effects of the proposed balancing and feature selection procedures. Evaluating cross-day generalization using the original CICDDoS2019 train-day and test-day partitions remains an important direction for future work.

Train-Test Split A stratified train-test split was performed using a 70/30 ratio, where 70% of the instances were allocated for training and 30% were reserved for testing. Stratification was employed to preserve the original class distribution across all attack categories and benign traffic, ensuring representative samples in both subsets despite the severe class imbalance present in the dataset. The testing subset remained completely isolated during model training, feature selection, and balancing operations, thereby preventing information leakage and enabling an unbiased evaluation on unseen samples. Table 3 summarizes the resulting dataset partitions.

Table 3. Train-Test Split Statistics

Dataset	Number of Instances
Training Set	1,547,787
Testing Set	663,338

Feature Scaling Feature scaling was applied to normalize feature values and reduce the influence of differing feature magnitudes during model training. Two commonly used scaling techniques, MinMaxScaler and StandardScaler, were evaluated during preliminary experiments. StandardScaler consistently produced superior classification performance across the evaluated models and was therefore adopted in the proposed framework.

To prevent information leakage, scaling parameters were learned exclusively from the training data and subsequently applied to the testing data using the same transformation. The standardized datasets were then used as input to the subsequent class balancing and feature selection stages.

Imbalance Dataset Treatment As shown in Table 1, the original dataset suffers from severe class imbalance, which can lead to biased model learning and poor generalization on minority attack classes. To mitigate this issue, we adopted a threshold-based class balancing strategy using Random Under Sampling (RUS)³⁶ and the Synthetic Minority Over-sampling Technique (SMOTE)^{41–43}. A hybrid balancing strategy was selected because exclusive under-sampling may discard substantial amounts of useful information from majority classes, whereas exclusive over-sampling may introduce excessive synthetic samples. Combining RUS and SMOTE enables more controlled balancing while preserving representative information from both majority and minority classes.

To investigate the impact of class balancing intensity on both detection performance and computational efficiency, three balanced training datasets were generated using statistical thresholds derived from the class distribution, namely the minimum, median, and mean class frequencies. These thresholds represent progressively increasing levels of data retention and balancing strictness, enabling a systematic evaluation of the proposed framework under different class distribution scenarios while maintaining a reproducible balancing strategy. The resulting dataset statistics are summarized in Table 4, and the construction logic is defined in Equations 8–10.

Table 4. Train Dataset(s) Statistics

Train Dataset(s)	Number of Instances
Dataset-I (Min)	2,120
Dataset-II (Median)	177,712
Dataset-III (Mean)	1,547,784

All train-test splitting, RUS, and SMOTE operations were performed using a fixed random seed of 42. This configuration ensures that the balancing process and experimental results can be reproduced consistently.

Dataset-I (Min Threshold) Let $X = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$ be the original dataset, where x_i is a feature vector and $y_i \in \mathcal{C}$ is the class label from the set of all classes $\mathcal{C}$. For each class $c \in \mathcal{C}$, let N_c denote the number of instances belonging to class c . The minimum class size is defined as

$$N_{\min} = \min_{c \in \mathcal{C}} N_c \quad (8)$$

Dataset-I is constructed by randomly under-sampling each class to contain exactly $N_{\min}$ instances. This dataset represents an aggressively balanced scenario in which all classes contribute equally using the smallest available class size. Although this substantially reduces the amount of training data, it provides an important reference point for evaluating the effects of severe class balancing and maximum dataset reduction.

Dataset-II (Median Threshold) The median class size is defined as

$$N_{\text{med}} = \text{median}(\{N_c \mid c \in \mathcal{C}\}) \quad (9)$$

Dataset-II is generated using a hybrid balancing strategy. Classes with $N_c > N_{\text{med}}$ are under-sampled, whereas classes with $N_c < N_{\text{med}}$ are over-sampled using SMOTE until each class contains N_{med} instances. This dataset represents a moderately balanced scenario that preserves substantially more training information than Dataset-I while maintaining class uniformity.

Dataset-III (Mean Threshold) The mean class size is defined as

$$N_{\text{mean}} = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} N_c \quad (10)$$

Dataset-III is constructed by adjusting all classes to contain N_{mean} instances through a combination of RUS and SMOTE. Compared with the previous datasets, this strategy retains the largest amount of training information and therefore represents the most conservative balancing scenario.

The objective of constructing these three balanced datasets is not to reproduce a specific operational traffic distribution but rather to evaluate the robustness of the proposed RO-DAD framework under varying levels of class imbalance correction. Dataset-I represents aggressive balancing, Dataset-II represents moderate balancing, and Dataset-III represents conservative balancing. Evaluating the framework across these datasets provides insight into the trade-off between classification performance, feature reduction effectiveness, and computational efficiency under different balancing conditions.

Domain Knowledge (DK) Filter The first stage of the proposed Resource-Optimized DDoS Attack Detection (RO-DAD) framework applies a Domain Knowledge (DK) filter to reduce the dimensionality of the dataset by eliminating features that are unlikely to contribute meaningfully to DDoS attack detection. This filtering stage is motivated by the observation that certain network traffic attributes represent administrative information, flow identifiers, or environment-specific metadata rather than behavioral characteristics associated with attack activity. Retaining such features may increase computational overhead, introduce unnecessary complexity, and potentially reduce the generalizability of the resulting detection models.

Feature elimination at this stage is guided by networking domain knowledge and findings reported in prior DDoS detection studies¹². Rather than relying solely on statistical criteria, the DK filter incorporates semantic understanding of network traffic features to identify attributes that provide limited discriminatory value for distinguishing benign and malicious traffic. Examples include identifiers, addressing information, and other metadata whose values are specific to a particular capture environment and may not generalize across different deployment scenarios.

The DK filter is applied uniformly across all experimental datasets before any statistical feature analysis is performed. By removing obviously irrelevant or non-informative attributes at an early stage, the framework reduces the search space for subsequent feature selection procedures, thereby improving computational efficiency and simplifying the learning process. The resulting reduced feature set serves as the input to the Least Significant (LS) filter, which further refines the feature space using statistical analysis.

Least Significant (LS) Filter Following the Domain Knowledge (DK) filtering stage, the Least Significant (LS) filter is employed as the second step in the proposed RO-DAD detection pipeline. While the DK filter removes environment-dependent or semantically irrelevant features based on expert insight, the LS filter performs variance-based feature elimination by identifying and removing features with zero variance across the training data. Such features are non-informative for classification tasks and do not contribute to discriminative learning. By excluding them, we reduce feature dimensionality and computational overhead, thereby supporting improved model efficiency. In this layer, 12 such features were discarded, as shown in Table 5, ensuring that only informative and statistically meaningful inputs are forwarded to the subsequent stages. The effectiveness of variance-based feature filtering in intrusion detection systems has been demonstrated in prior studies^{44–46}.

Table 5. Features Removed by the Least Significant (LS) Filter

Sr. No.	Feature Name
1	Bwd_PSH_Flags
2	Fwd_URG_Flags
3	Bwd_URG_Flags
4	FIN_Flag_Count
5	PSH_Flag_Count
6	ECE_Flag_Count
7	Fwd_Avg_Bytes/Bulk
8	Fwd_Avg_Packets/Bulk
9	Fwd_Avg_Bulk_Rate
10	Bwd_Avg_Bytes/Bulk
11	Bwd_Avg_Packets/Bulk
12	Bwd_Avg_Bulk_Rate

Most Significant (MS) Filter As the third stage of the proposed RO-DAD framework, following the Domain Knowledge (DK) and Least Significant (LS) filters, the Most Significant (MS) filter is designed to identify highly informative features using model-driven feature importance analysis. Tree-based algorithms, including DT, RF, and XGBoost, are particularly suitable for high-dimensional data as they inherently estimate feature importance during training. These importance scores quantify the contribution of each feature to the predictive performance of the model through measures such as information gain and impurity reduction.

To perform feature selection, a threshold-based criterion is applied as defined in Equation 11:

$$S = \{f_i \mid I(f_i) > \tau\} \quad (11)$$

where f_i denotes a feature from the original feature set, $I(f_i)$ represents its importance score, and τ is the minimum importance threshold. Features satisfying $I(f_i) > \tau$ are retained, while the remaining features are discarded.

To determine an appropriate threshold, a sensitivity analysis was conducted using multiple values of $\tau \in \{0.005, 0.01, 0.02, 0.05\}$. For each setting, DT, RF, and XGBoost classifiers were trained using the resulting feature subsets, and their weighted F1-scores along with the number of retained features were recorded. The results are summarized in Table 6 and Figure 2.

Table 6. Sensitivity Analysis of Feature Importance Threshold (τ)

τ	Features (DT/RF/XGB)	DT F1	RF F1	XGB F1
0.005	10 / 30 / 13	0.9902	0.9917	0.9904
0.010	10 / 28 / 12	0.9902	0.9917	0.9903
0.020	5 / 16 / 8	0.9863	0.9906	0.9845
0.050	3 / 8 / 5	0.9844	0.9826	0.9717

The results indicate a stable performance region between $\tau = 0.005$ and $\tau = 0.01$, where all three classifiers maintain nearly identical weighted F1-scores. However, increasing the threshold beyond $\tau = 0.01$ significantly reduces the number of selected features and leads to a gradual decline in classification performance. For instance, in the case of the DT model, increasing τ from 0.01 to 0.05 reduces the feature set from 10 to 3 features and decreases the F1-score from 0.9902 to 0.9844, with similar trends observed for RF and XGBoost.

Based on this trade-off between dimensionality reduction and predictive performance, $\tau = 0.01$ is selected as the optimal threshold, as it represents the highest value before noticeable performance degradation occurs. This choice ensures effective feature reduction while preserving classification accuracy.

Because tree-based models compute feature importance differently, feature rankings were generated independently using DT, RF, and XGBoost to assess consistency across models. Table 7 presents the top-ranked features obtained from each model. A substantial overlap is observed among the selected features, including *Min_pkt_len*, *Average_pkt_Size*, *Fwd_pkt_len_Mean*, and *Max_pkt_len*, indicating stable feature relevance across different learning algorithms.

Among the evaluated models, RF provides the most balanced representation of informative features, retaining a comprehensive subset that overlaps strongly with both DT and XGBoost selections while maintaining stability across threshold variations. Therefore, RF is selected as the final feature selector for the MS stage. The RF-derived feature subset is used as the final

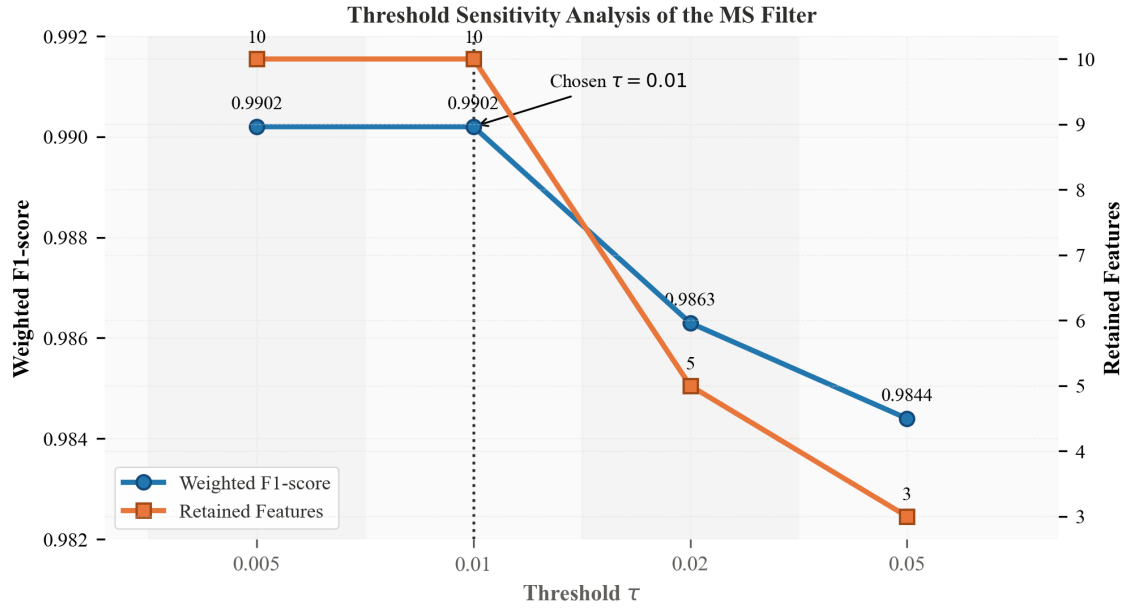


Figure 2. Sensitivity analysis of feature importance threshold (τ) across DT, RF, and XGBoost models showing weighted F1-score variation and feature reduction behavior.

output of the RO-DAD feature selection pipeline, while DT and XGBoost subsets are retained for comparative evaluation and validation.

The effectiveness of model-based feature importance techniques for reducing dimensionality while maintaining classification performance has been widely reported in recent intrusion detection studies^{44,45,47}.

Model Training Multiple ML models are trained on the refined dataset obtained after feature selection to learn discriminative patterns for DDoS attack detection. The performance of the proposed feature selection framework is assessed using traditional baseline classifiers like LR, SVM, GNB, and DT as well as ensemble-based classifiers like RF, AdaBoost, XGBoost and stacking-based classifiers. These models are popular in intrusion detection studies because they can represent linear and non-linear relationships between network traffic data.

Model Validation The trained models are then tested on a test set that was not used for training to check their generalization ability. The classification results are evaluated based on the standard classification measures on data preprocessed with the same parameters obtained from the training set. In particular, feature scaling is performed with parameters learned from training data, and applied to test data without data leakage and to fairly evaluate the performance.

These stages enhance the effectiveness of RO-DAD framework by increasing the accuracy of detection, robustness, and generalization capability of DDoS attack detection in network environments.

Results and Discussion

This section presents the experimental results and evaluates the performance of the proposed DDoS detection framework. Hardware specifications and implementation details are summarized in Table 8.

Experimental Setup

Experimental Configuration A random seed of 42 was applied throughout all experiments to ensure consistent and reproducible outcomes, wherever the learning algorithm permitted. A stratified 70/30 split was applied to divide the dataset into training and testing subsets, ensuring that class proportions remained balanced across both portions. The ML models were configured as follows:

- Decision Tree (DT): entropy criterion, maximum depth of 15, minimum samples split of 10, and minimum samples leaf of 2.
- Random Forest (RF): 300 trees, entropy criterion, maximum depth of 15, and minimum samples split of 10.

Table 7. Comparison of Selected Features with Tree-based ML Models

Sr.No.	Decision Tree (DT)	XGBoost	Random Forest (RF)
1	Fwd_pkt_len_Mean	Idle_Std	Fwd_pkt_len_Min
2	Average_pkt_Size	Bwd_pkt_len_Max	Min_pkt_len
3	Min_pkt_len	Fwd_pkt_len_Std	Fwd_pkt_len_Mean
4	Fwd_pkt_len_Max	URG_Flag_Count	Average_pkt_Size
5	Flow_IAT_Std	Min_pkt_len	Avg_Fwd_Segment_Size
6	Fwd_Header_len	Average_pkt_Size	pkt_len_Mean
7	Flow_IAT_Mean	Init_Win_bytes_fwd	Fwd_pkt_len_Max
8	Fwd_IAT_Min	Max_pkt_len	Max_pkt_len
9	Flow_Bytes/s	Fwd_pkt_len_Mean	Subflow_Fwd_Bytes
10	Flow_IAT_Min	Flow_IAT_Std	Flow_IAT_Mean
11	Max_pkt_len	Idle_Min	Total_len_of_Fwd_pkts
12		Fwd_pkt_len_Min	Flow_Bytes/s
13		ACK_Flag_Count	pkt_len_Variance
14		min_seg_size_fwd	Flow_pkts/s
15		Fwd_pkts/s	pkt_len_Std

Abbreviations: Packet (pkt), Length (len), forward (fwd)

Table 8. Hardware and Implementation Details

Component	Description
LAPTOP MODEL	HP ProBook 450 G8
PROCESSOR	11th Gen Intel(R) Core (TM) i7-1165G7 @ 2.80GHz
RAM	32GB
OPERATING SYSTEM	Windows 11
PROGRAMMING LANGUAGE	Python
DEVELOPMENT ENVIRONMENT	Jupyter Notebook, Microsoft Visual Studio Code
KEY LIBRARIES	Scikit-learn, NumPy, Pandas, imbalanced-learn (SMOTE)

- Logistic Regression (LR): multinomial classification with Newton-CG solver and maximum iteration limit of 1000.
- Support Vector Machine (SVM): one-versus-one (OvO) decision strategy using default kernel-based optimization.
- AdaBoost: default implementation parameters.
- Gaussian Naive Bayes (GNB): default implementation parameters.
- XGBoost (XGB): multi-class soft probability objective (multi:softprob) with default parameters unless otherwise specified.
- Stacked Ensemble: base learners consisting of DT, RF, and Gradient Boosting classifiers. XGBoost was used as the meta-classifier with five-fold cross-validation.

Implementation Details All experiments were implemented in Python using standard ML and data processing libraries. The key library versions used to ensure reproducibility are listed below:

- Python: 3.10.16
- Scikit-learn: 1.6.1
- NumPy: 1.26.4
- Pandas: 2.2.3
- Imbalanced-learn: 0.13.0

No GPU acceleration was used for baseline ML models. Tree-based ensemble models were optimized using CPU parallelization where applicable.

Datasets and Evaluation Strategy

We evaluate the robustness of the proposed framework across multiple DDoS attack datasets generated through different class balancing strategies. Multiple ML models are used for evaluation. Eight models are evaluated in Dataset-I and Dataset-II, while six models are evaluated in Dataset-III due to computational constraints. The experimental analysis focuses on assessing the effectiveness and computational efficiency of the proposed Resource-Optimized DDoS Attack Detection framework in realistic scenarios.

Baseline Feature Set Construction

In the DK-Filter approach, we identified and removed features that are environment-dependent and do not contribute to the training of ML models. The removal of these features ensures that the dataset consists only of the most relevant features. We declared the output of the DK-Filter as the baseline feature set used in all experiments. This provides a consistent and realistic benchmark for evaluating the proposed framework, ensuring that improvements are directly attributable to the proposed feature optimization strategy.

Results on Dataset-I

Dataset-I was created by balancing class instances to the minimum value among all classes, ensuring equal representation and addressing class imbalance. Eight ML models were assessed, using accuracy, precision, recall, and F1-score.

Performance Analysis

Figure 3 compares weighted F1-scores of baseline and RO-DAD framework. The proposed method consistently improves or maintains performance across most models, particularly in imbalanced DDoS attack detection scenarios. This highlights the effectiveness of the proposed feature selection strategy in enhancing detection accuracy while supporting diverse ML models.

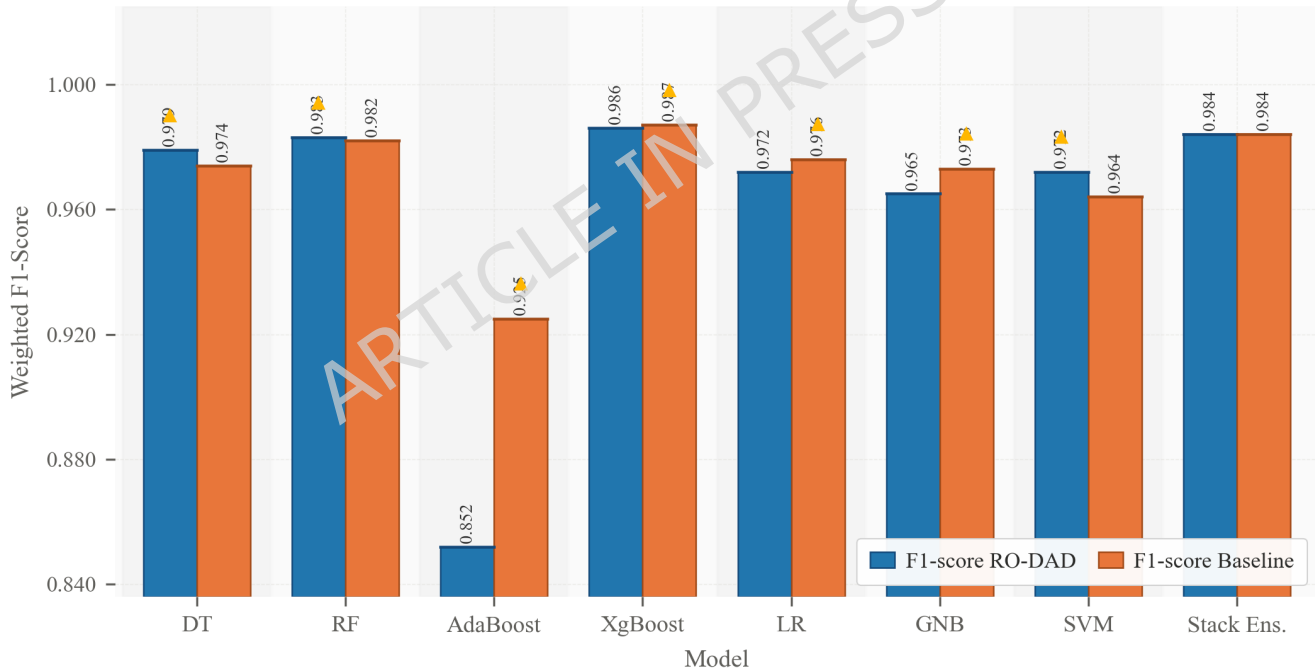


Figure 3. Comparison of Weighted F1-Scores for ML Models Using the Proposed RO-DAD Framework and Baseline Model

Table 9 summarizes classification performance across all evaluated ML models. Ensemble-based models, particularly RF and XGBoost, achieved the highest overall performance, demonstrating strong robustness and detection capability. DT and SVM also exhibited consistent performance, while LR and GNB showed balanced results. AdaBoost performed comparatively lower, indicating sensitivity to feature reduction. Overall, the results confirm the effectiveness of the proposed framework.

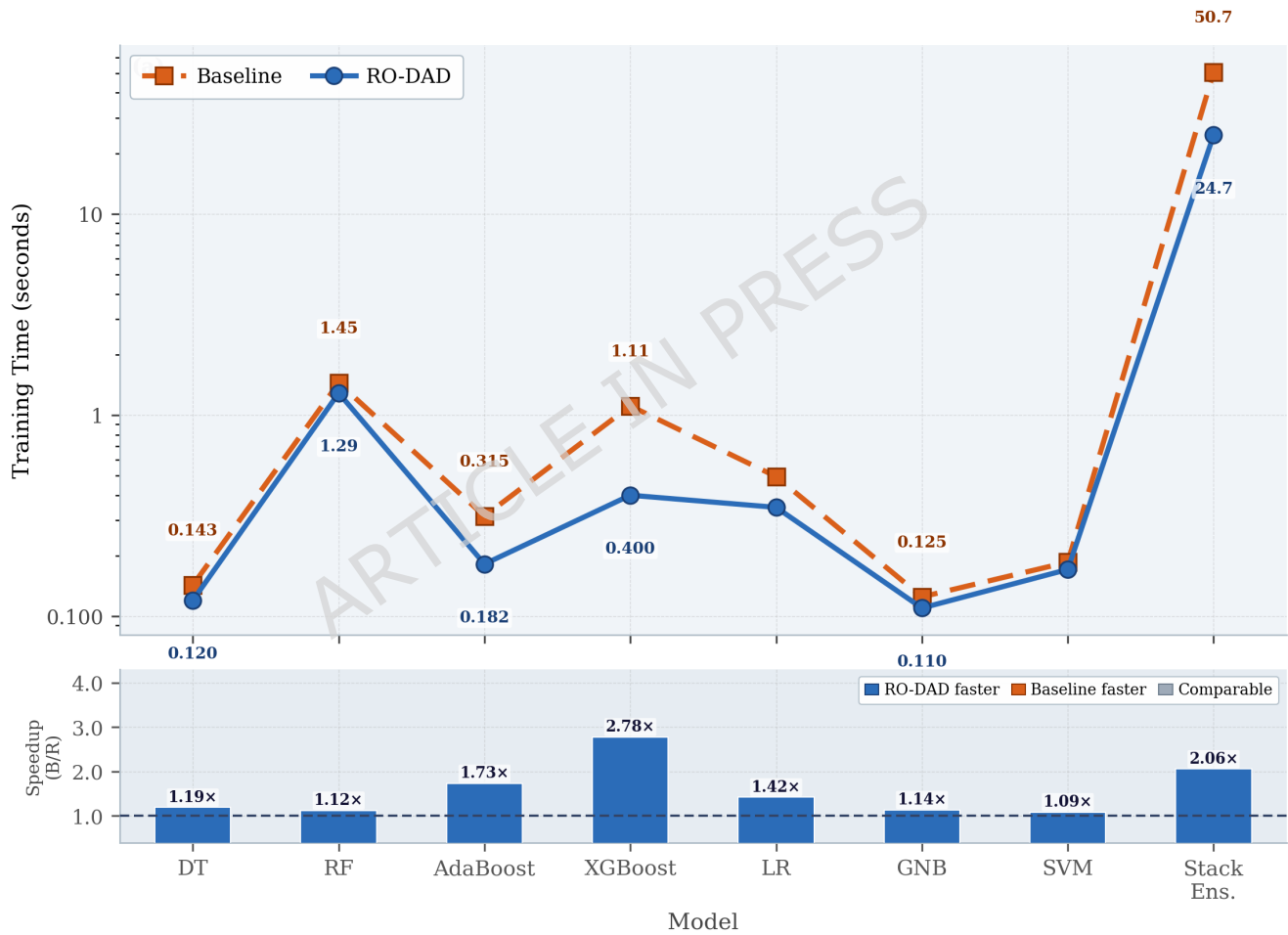
Resource Utilization Analysis

To further evaluate the efficiency of the proposed framework, resource utilization is analyzed in terms of training time, prediction time, and memory consumption. These metrics provide a practical measure of computational cost in addition to predictive performance. Figure 4 compares training times between baseline and RO-DAD framework. The proposed framework reduces

Table 9. Summary of proposed RO-DAD Framework Performance Across ML Models

ML Model	Accuracy	Precision	Recall	F1-score
DT	0.973	0.987	0.973	0.979
RF	0.978	0.989	0.978	0.983
AdaBoost	0.848	0.863	0.848	0.852
XGBoost	0.982	0.991	0.982	0.986
LR	0.960	0.987	0.960	0.972
GNB	0.958	0.978	0.958	0.965
SVM	0.962	0.986	0.962	0.972
Stacked Ensemble	0.978	0.991	0.978	0.984

training time across all evaluated models. The reduction ranges from 8.08% for SVM to 63.95% for XGBoost, demonstrating the impact of feature optimization on model training efficiency.

**Figure 4.** Comparison of Training Times for ML Models Using the Proposed RO-DAD Framework and Baseline Model

Similarly, Figure 5 presents the prediction time comparison between the baseline and RO-DAD frameworks. Prediction time reductions range from 16.98% for SVM to 50.20% for GNB, indicating that the optimized feature subset improves inference efficiency while maintaining detection performance.

Table 10 presents the comparison of training memory consumption between the baseline and the proposed RO-DAD framework. The results indicate that most ML models require lower memory usage after applying the optimized feature selection strategy. Significant reductions are observed in SVM, LR, and DT, confirming reduced computational overhead.

Overall, the proposed RO-DAD framework demonstrates improved computational efficiency by reducing training time,

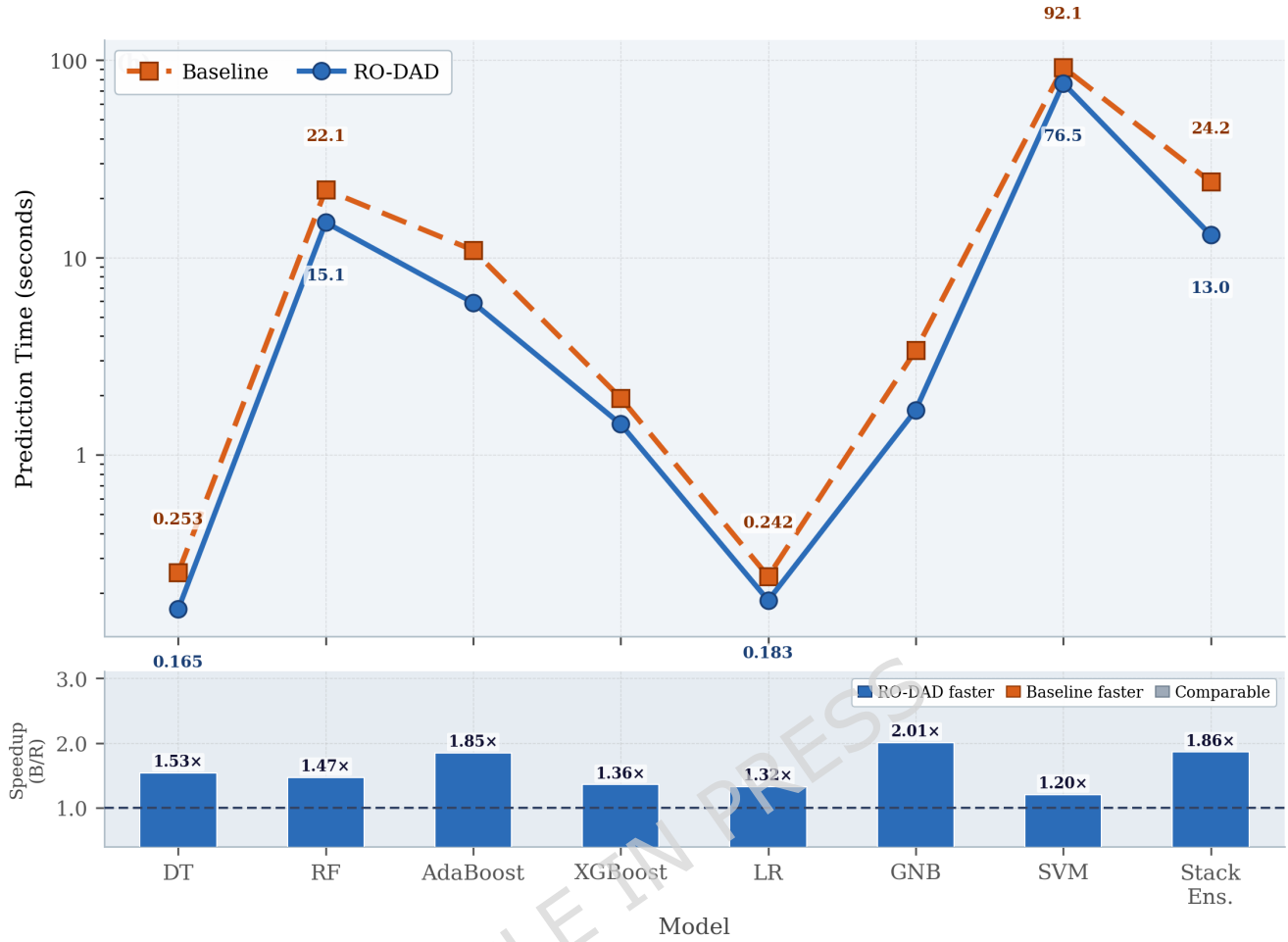


Figure 5. Comparison of Prediction Times for ML Models Using the Proposed RO-DAD Framework and Baseline Model

Table 10. Memory Consumption Comparison with Percentage Reduction (Dataset-I)

ML Model	Baseline (MB)	RO-DAD (MB)	Reduction (%)
DT	0.1646	0.0273	83.45%
RF	5.0938	4.0273	20.94%
LR	0.1953	0.0117	94.02%
AdaBoost	0.0391	0.0273	30.18%
SVM	0.6445	0.0625	90.37%
XGBoost	0.1133	0.0977	13.77%

prediction time, and memory consumption while maintaining strong DDoS attack detection performance. These findings highlight the suitability of the proposed framework for practical and resource-optimized intrusion detection applications.

Results on Dataset-II

Dataset-II was generated by balancing class instances to the median value among all classes. The median-based balancing strategy provides a moderate level of class representation while reducing the impact of class imbalance. The same eight ML models were evaluated on Dataset-II. The objective was to assess the effectiveness of the proposed RO-DAD framework under a different class distribution scenario and to examine its robustness across varying dataset balancing strategies.

Performance Analysis

Figure 6 compares the weighted F1-scores achieved using baseline features and the proposed RO-DAD framework on Dataset-II. The results show that the proposed framework consistently maintains or improves classification performance across most ML

models. RF, DT, XGBoost, and the Stacked Ensemble consistently achieve strong performance, indicating that the optimized feature selection strategy effectively preserves discriminative information while reducing feature dimensionality. Although AdaBoost exhibits a performance decrease, the overall trend confirms the robustness of the proposed framework across different learning algorithms.

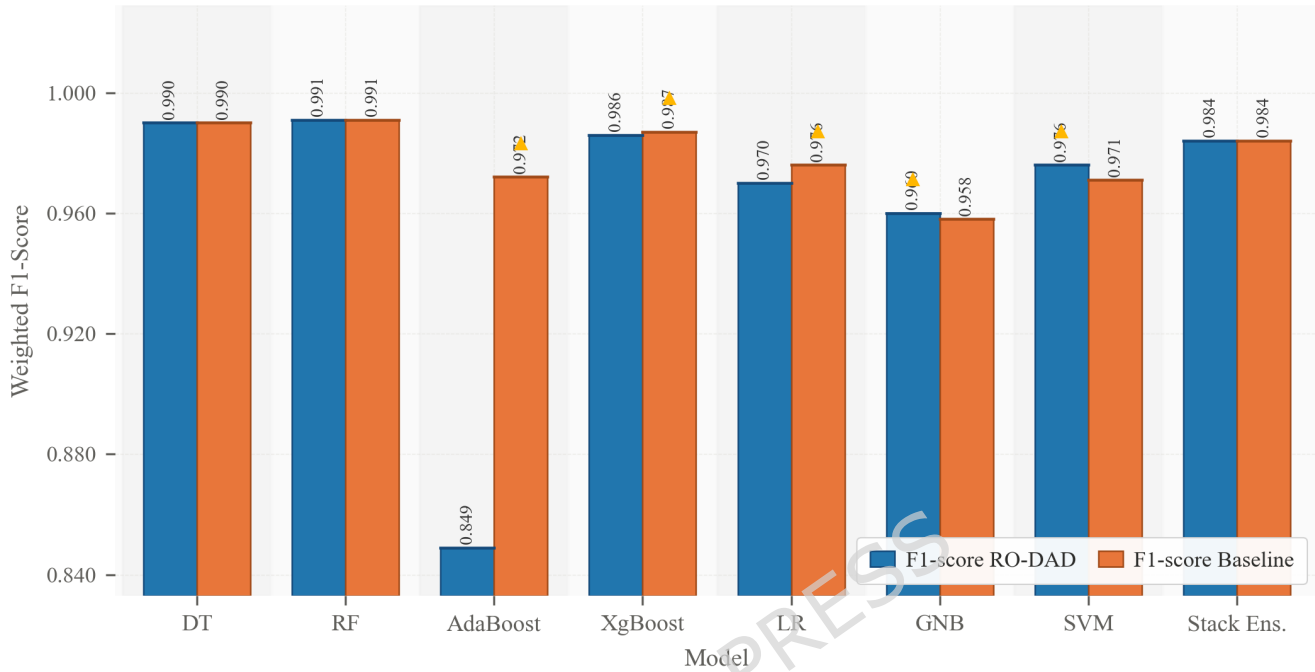


Figure 6. Comparison of Weighted F1-Scores for ML Models Using the Proposed RO-DAD Framework and Baseline Model

Table 11 summarizes the classification performance achieved by the proposed framework on Dataset-II. RF achieved the highest overall performance with an accuracy of 0.989 and weighted F1-score of 0.991. DT and XGBoost also demonstrated strong performance, achieving weighted F1-scores of 0.990 and 0.986, respectively. The Stacked Ensemble model maintained competitive performance, while LR, GNB, and SVM produced stable and consistent results. AdaBoost achieved comparatively lower performance, suggesting greater sensitivity to feature reduction and class distribution changes. Overall, the results confirm the effectiveness of the proposed framework across a moderately balanced dataset.

Table 11. Summary of proposed RO-DAD framework performance across ML models

ML Model	Accuracy	Precision	Recall	F1-score
Decision Tree (DT)	0.987	0.993	0.987	0.990
Random Forest (RF)	0.989	0.994	0.989	0.991
AdaBoost	0.814	0.973	0.814	0.849
XGBoost	0.982	0.991	0.982	0.986
Logistic Regression (LR)	0.954	0.989	0.954	0.970
Gaussian NB (GNB)	0.950	0.980	0.950	0.960
Support Vector Machine (SVM)	0.968	0.987	0.968	0.976
Stacked Ensemble	0.978	0.991	0.978	0.984

Resource Utilization Analysis

Figures 7 and 8 compare the training and prediction times of the baseline and proposed frameworks for Dataset-II. Similar to Dataset-I, the RO-DAD framework reduces computational overhead across most evaluated models. The observed reductions in both training and inference time indicate that the optimized feature subset contributes to more efficient model execution without sacrificing classification performance. These results further support the practical applicability of the proposed framework in cost-sensitive and large-scale cloud environments.

Following the evaluation protocol defined in Dataset-I, Dataset-II focuses on training and prediction time analysis to

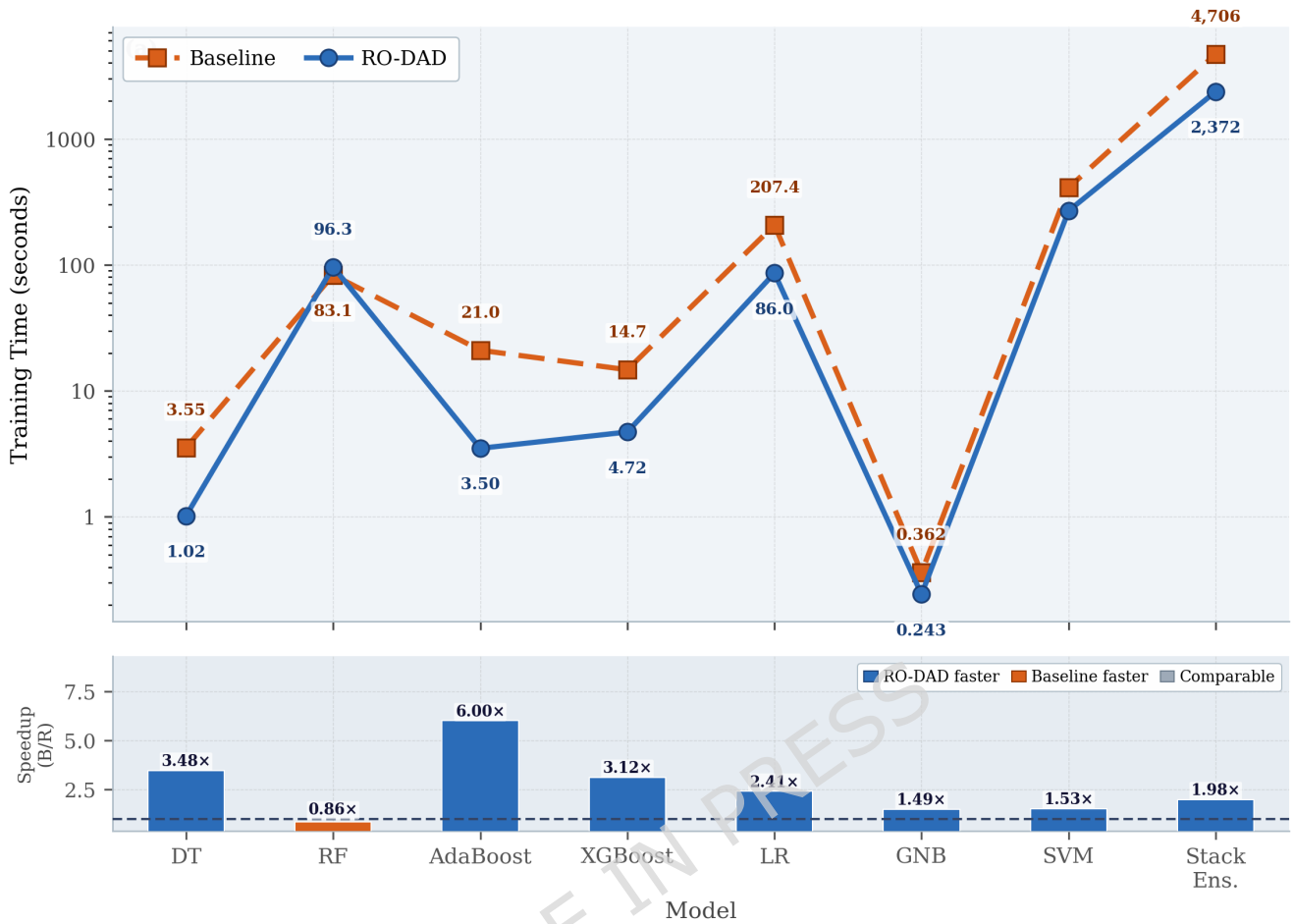


Figure 7. Comparison of Training Times for ML Models Using the Proposed RO-DAD Framework and Baseline Model

assess computational efficiency under a median-balanced class distribution. Overall, the results obtained from Dataset-II demonstrate that the proposed RO-DAD framework maintains strong detection performance across multiple ML models under a median-balanced class distribution. The consistent results indicate that the framework remains robust despite changes in class representation and balancing strategy.

Results on Dataset-III

Finally, we present the experimental results for Dataset-III. This dataset was generated by balancing class instances to the mean value across all classes. The mean based balancing strategy provides an alternative perspective on handling class imbalance by increasing the representation of minority classes while retaining a larger overall dataset size. Due to the substantially larger size of Dataset-III, SVM and Stacked Ensemble models were excluded because of their computational requirements. Six ML models were evaluated on Dataset-III to assess the effectiveness of the proposed RO-DAD framework under this balancing strategy. The resulting performance metrics provide additional insight into the robustness and scalability of the proposed framework for DDoS attack detection.

Performance Analysis

Figure 9 compares the weighted F1-scores obtained using baseline features and the proposed RO-DAD framework on Dataset-III. The results show that the proposed framework consistently maintains or improves classification performance across most evaluated models. DT and RF continue to achieve strong performance, while AdaBoost demonstrates noticeable improvement under the optimized feature set. These results indicate that the proposed feature selection strategy effectively preserves relevant information while reducing feature dimensionality, thereby supporting accurate DDoS attack detection.

Table 12 summarizes the classification performance achieved by the proposed framework on Dataset-III. RF achieved the highest overall performance with an accuracy of 0.990 and weighted F1-score of 0.992. DT also demonstrated excellent

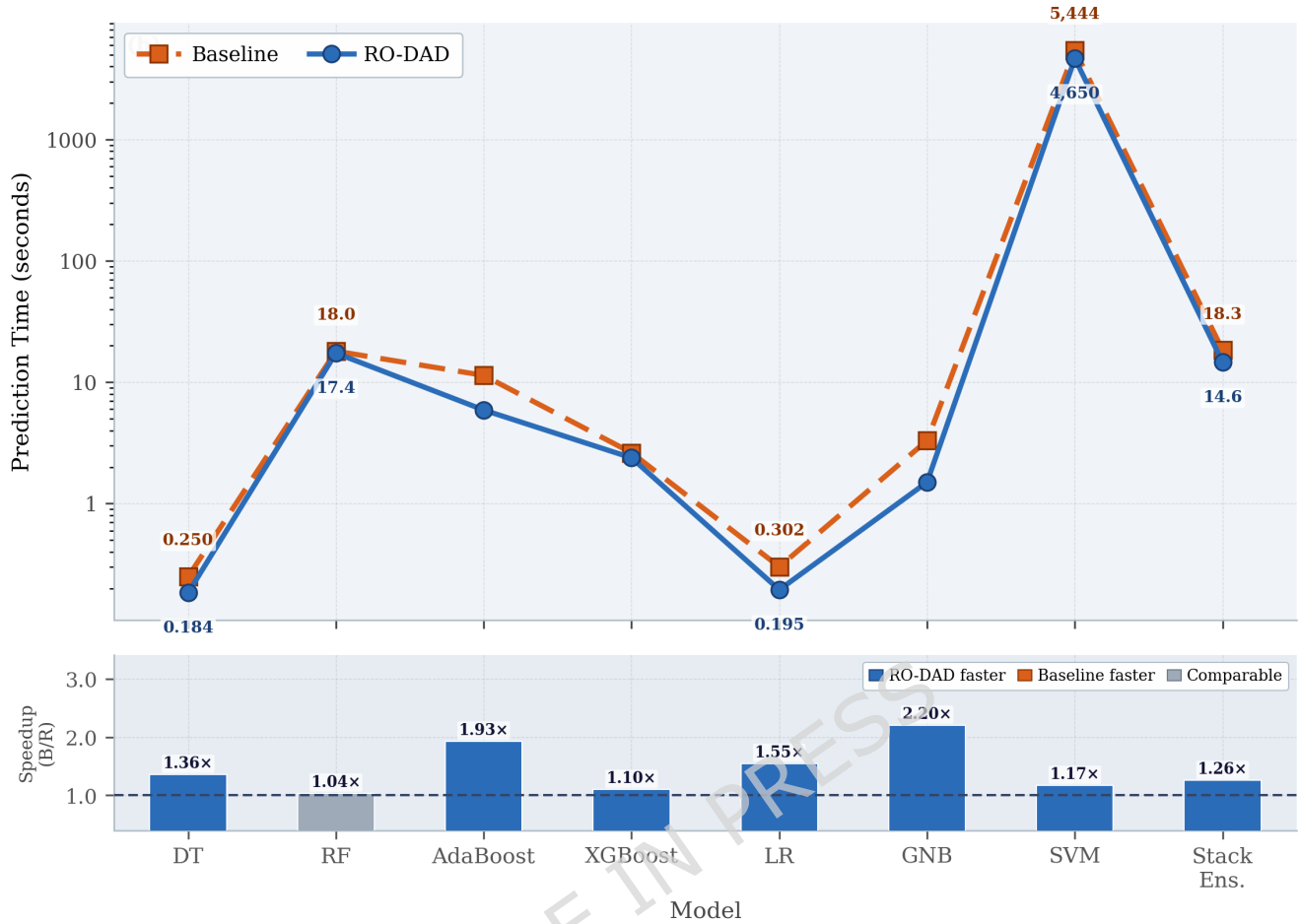


Figure 8. Comparison of Prediction Times for ML Models Using the Proposed RO-DAD Framework and Baseline Model

performance, achieving a weighted F1-score of 0.991. XGBoost maintained strong and consistent results, while LR and GNB produced balanced performance across all evaluation metrics. Although AdaBoost achieved comparatively lower performance than the leading models, it still maintained satisfactory detection capability. Overall, the results confirm the effectiveness of the proposed framework under the mean-balanced dataset configuration.

Table 12. Summary of proposed RO-DAD Framework Performance Across ML Models

ML Model	Accuracy	Precision	Recall	F1-score
Decision Tree (DT)	0.989	0.994	0.989	0.991
Random Forest (RF)	0.990	0.994	0.990	0.992
AdaBoost	0.975	0.973	0.975	0.973
XGBoost	0.982	0.991	0.982	0.986
Logistic Regression (LR)	0.954	0.989	0.954	0.971
Gaussian NB (GNB)	0.950	0.985	0.950	0.963

Resource Utilization Analysis

Figures 10 and 11 compare the training and prediction times of the baseline and proposed frameworks on Dataset-III. Similar to the observations from Dataset-I and Dataset-II, the RO-DAD framework reduces computational overhead for most evaluated ML models. The reduction in execution time demonstrates that the optimized feature subset contributes to more efficient model training and inference while maintaining strong classification performance. These findings further validate the scalability of the proposed framework on larger balanced datasets.

Consistent with the evaluation protocol, Dataset-III reports only training and prediction time metrics due to its larger scale

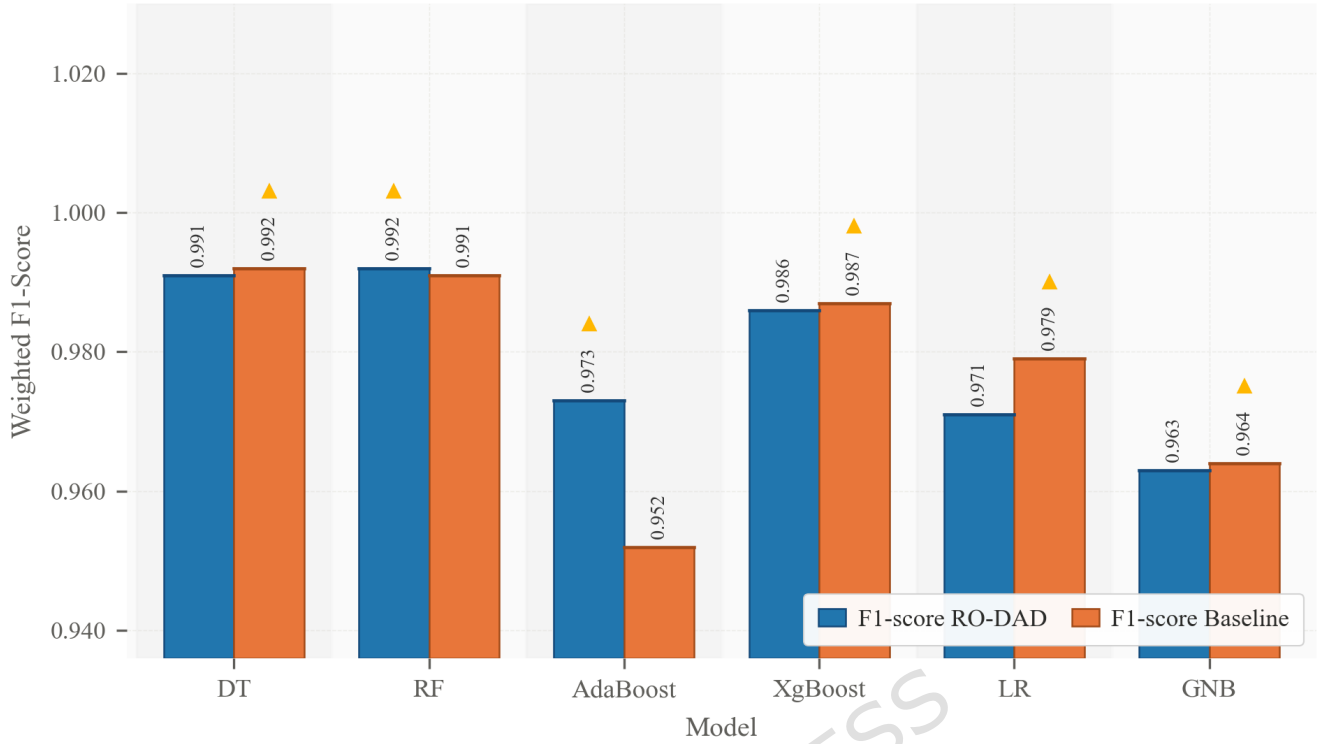


Figure 9. Comparison of Weighted F1-Scores for ML Models Using the Proposed RO-DAD Framework and Baseline Model

and computational cost constraints. Overall, the results obtained from Dataset-III demonstrate that the proposed RO-DAD framework maintains strong detection performance on a larger mean-balanced dataset. The results further validate the scalability and robustness of the framework while confirming its suitability for practical DDoS attack detection scenarios.

Cross Dataset Discussion

Across all three balancing strategies, the proposed RO-DAD framework consistently maintained high classification performance while preserving computational efficiency. RF and DT exhibited the most stable performance across all datasets, while XGBoost consistently ranked among the strongest performers. The reductions observed in training and prediction time indicate that the selected feature subset effectively lowers processing overhead without compromising detection capability. In addition, memory profiling on Dataset-I demonstrated reduced memory requirements during model training. Collectively, these findings suggest that the proposed framework remains robust across different class distributions and is not dependent on a specific balancing configuration.

Error Analysis and Class-wise Performance

To provide deeper insight beyond aggregate evaluation metrics, we perform a confusion-matrix-based error analysis along with class wise performance evaluation. For clarity, three representative models, DT, RF, and XGBoost are analyzed on Dataset-II, as they capture different learning paradigms. Similar trends are observed across the remaining models and datasets.

Figures 12, 13, and 14 present the confusion matrices for DT, RF, and XGB under both baseline and the proposed RO-DAD framework. The baseline models exhibit significant misclassification among minority classes, particularly Portmap and UDPLag, which are frequently misclassified as dominant traffic classes such as SYN and UDP. This highlights the effect of class imbalance in the baseline feature space.

In contrast, the proposed RO-DAD framework improves diagonal dominance across all models, indicating enhanced class separability. This improvement is particularly evident in minority class detection, where previously underrepresented attack types show higher correct classification rates. A detailed class-wise analysis shows that the recall for challenging classes such as UDPLag and Portmap is noticeably improved under the proposed framework, reducing false negatives and improving detection sensitivity. Furthermore, confusion between structurally similar classes, such as NetBIOS and Portmap, is noticeably reduced, indicating that the proposed feature selection strategy improves discriminative capability between closely related traffic patterns.

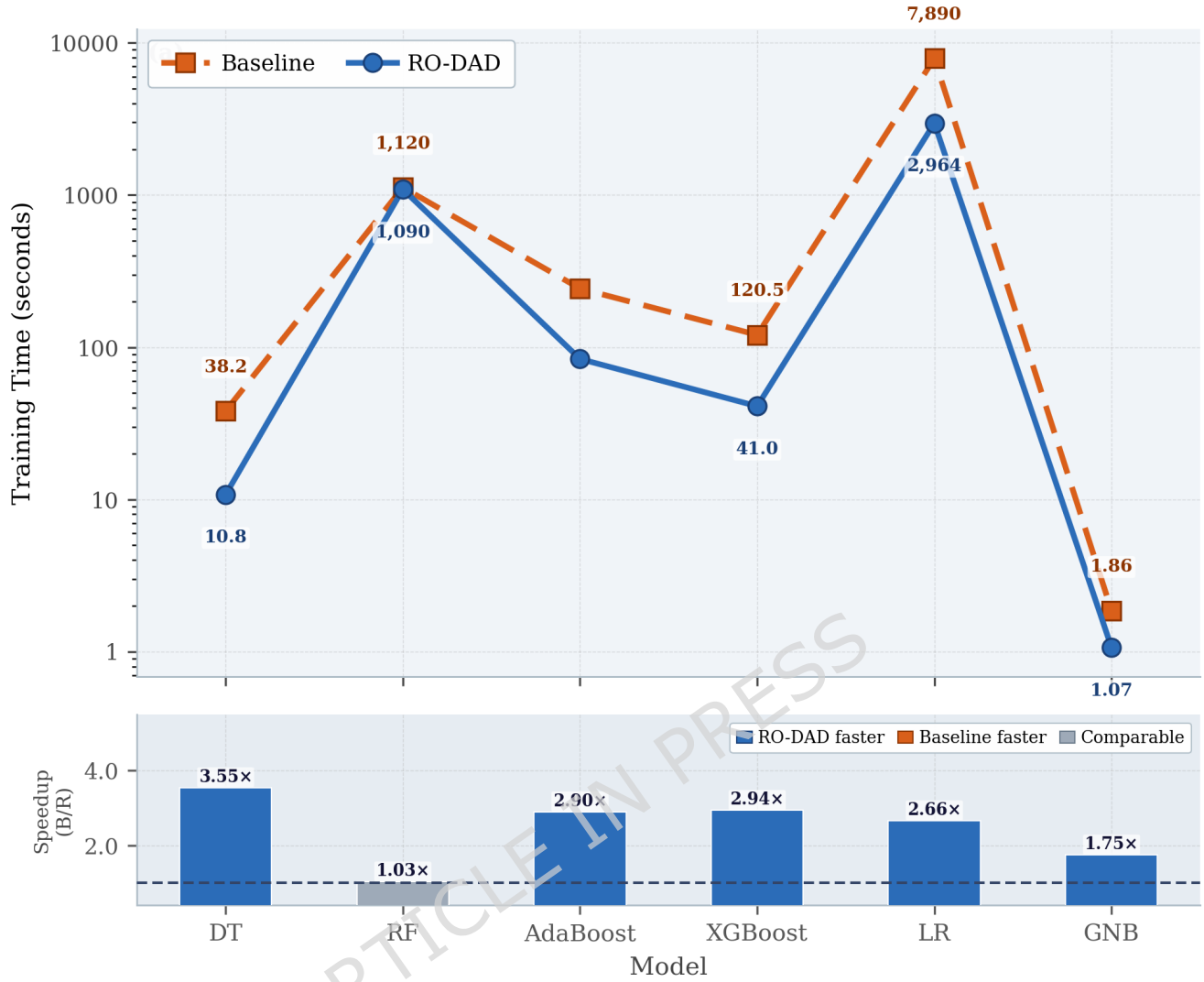


Figure 10. Comparison of Training Times for ML Models Using the Proposed RO-DAD Framework and Baseline Model

Overall, the RO-DAD framework demonstrates more balanced classification behavior across all evaluated models. While minor trade-offs are observed in some majority classes, these are offset by substantial improvements in minority class detection, resulting in improved robustness and generalization for imbalanced DDoS attack detection.

False Positive Rate Analysis

To further evaluate the operational reliability of the proposed framework, we analyze the False Positive Rate (FPR), which quantifies the proportion of benign instances incorrectly classified as attacks. This metric is particularly important in DDoS detection, where false alarms can significantly affect system usability and trust.

The macro-average FPR is computed from the confusion matrix for each model to provide an overall assessment of false alarm behavior. Table 13 summarizes the FPR values for both baseline and proposed frameworks across representative models.

The results indicate that the proposed RO-DAD framework maintains consistently low FPR across all evaluated models, comparable to the baseline approach. Specifically, RF shows a slight reduction in FPR, while DT and XGBoost exhibit minor variations within a very low range (on the order of 10^{-3}). These variations are negligible in practical terms and do not indicate any degradation in detection reliability. Importantly, the proposed framework preserves the false alarm behavior of the baseline while improving class-wise detection performance, particularly for minority attack classes such as Portmap and UDPLag. This demonstrates that the framework enhances detection capability without introducing additional false positives, which is critical for real-world intrusion detection systems. Overall, these results confirm that improvements in detection performance are achieved without compromising false alarm stability. The evaluation is further complemented by confusion matrix analysis,

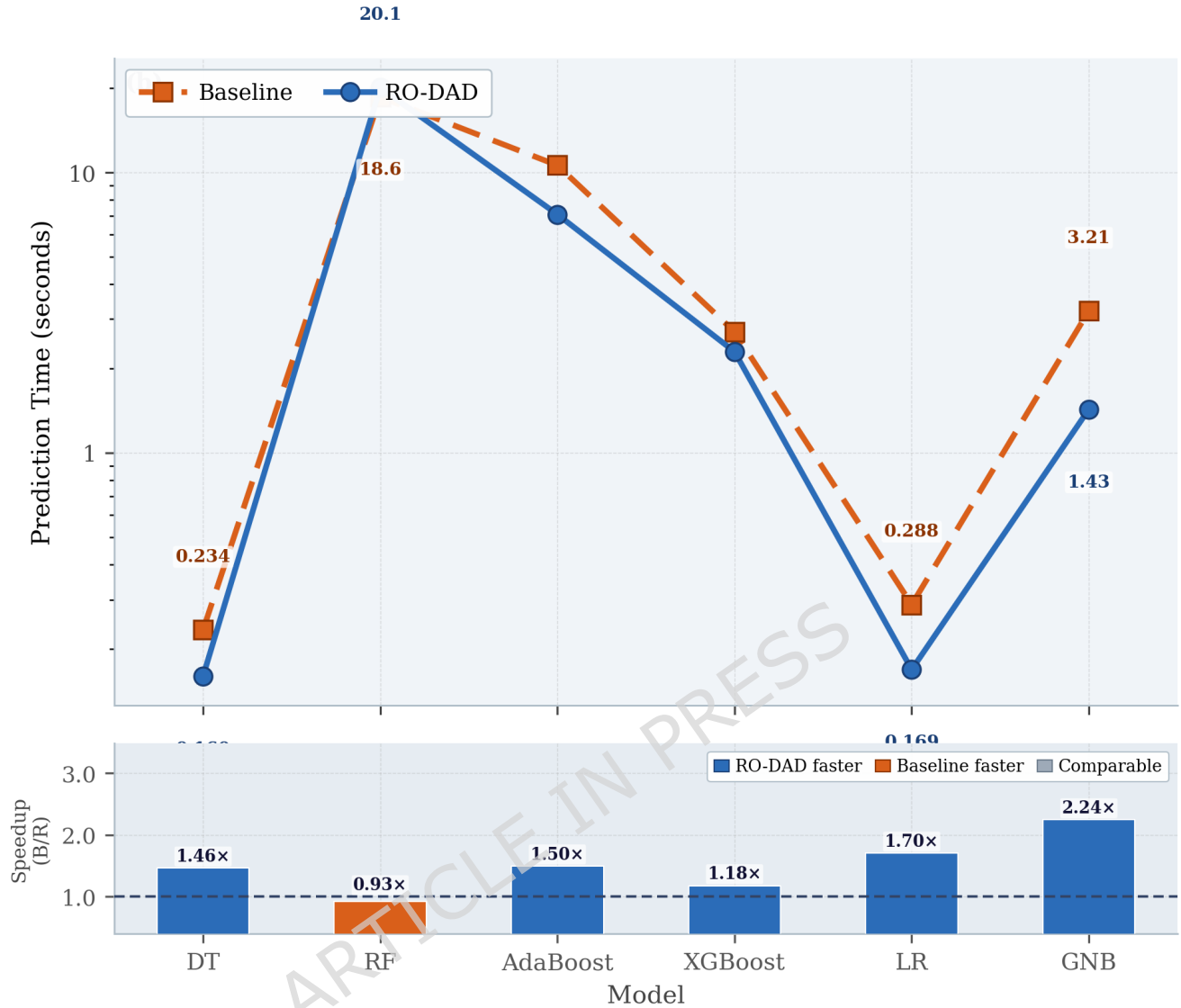


Figure 11. Comparison of Prediction Times for ML Models Using the Proposed RO-DAD Framework and Baseline Model

class-wise performance behavior, and false positive rate assessment, providing a comprehensive understanding of model performance beyond aggregate metrics.

ROC-AUC is not reported in this study due to the multi-class nature of the classification problem and the significant class imbalance present in the CICDDoS2019 dataset. In such imbalanced multi-class intrusion detection scenarios, confusion matrix-based analysis, macro-averaged F1-score, recall, and False Positive Rate (FPR) provide more reliable and interpretable evaluation compared to ROC-AUC. Although formal statistical significance tests such as the paired t-test or Wilcoxon signed-rank test were not conducted, the proposed framework was evaluated across three independently balanced datasets and multiple ML models. The observed improvements remained consistent across these experimental settings, suggesting that the performance gains are not attributable to a single dataset configuration. Furthermore, the confusion matrix analysis, class-wise performance evaluation, and false positive rate assessment collectively provide additional evidence of the robustness and repeatability of the proposed framework.

Comparative Analysis

Comparative Analysis with state of the art

The comparative analysis presented in Table 14 highlights the capabilities of various ML models used for detecting DDoS on the CICDDoS2019 dataset across multiple studies. Our proposed framework achieved competitive performance compared to

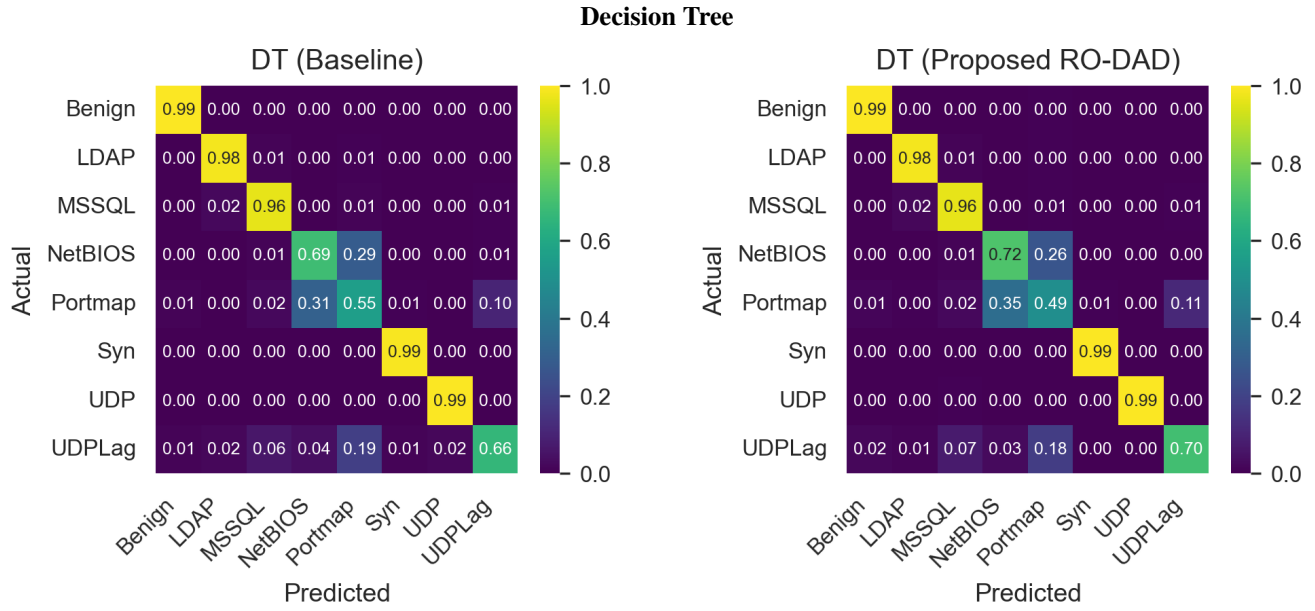


Figure 12. Confusion matrix comparison for Decision Tree under baseline and RO-DAD framework.

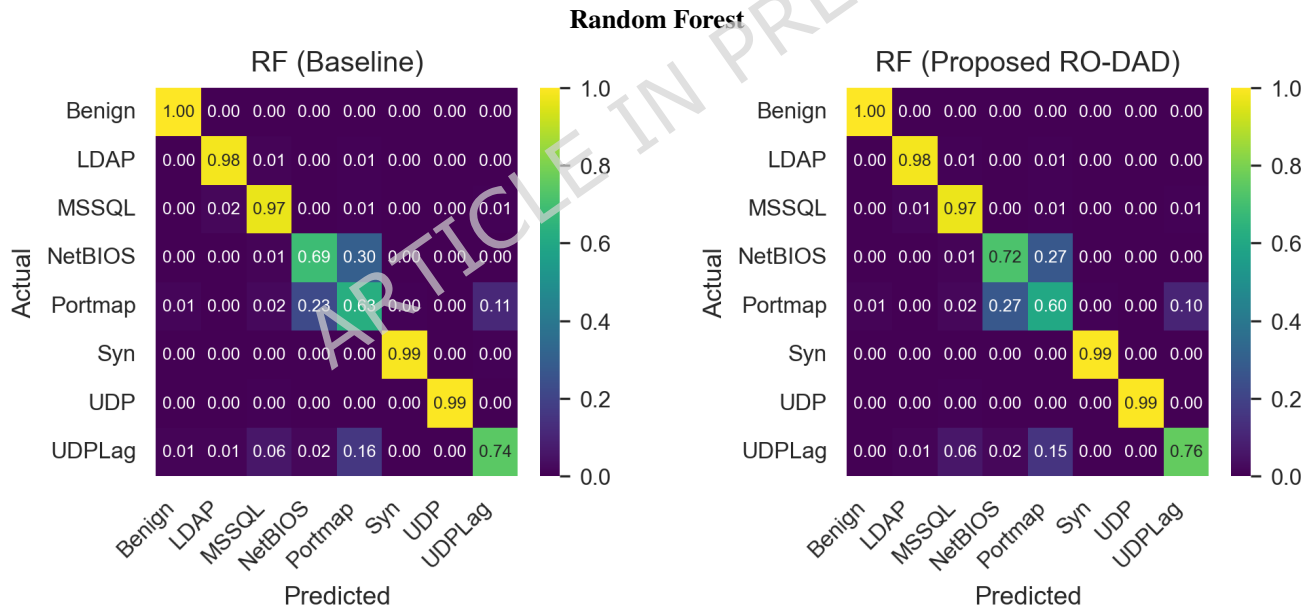


Figure 13. Confusion matrix comparison for Random Forest under baseline and RO-DAD framework.

Table 13. Macro-average False Positive Rate (FPR) Comparison

Model	Baseline FPR	Proposed FPR
Decision Tree (DT)	0.00162	0.00163
Random Forest (RF)	0.00154	0.00146
XGBoost	0.00212	0.00235

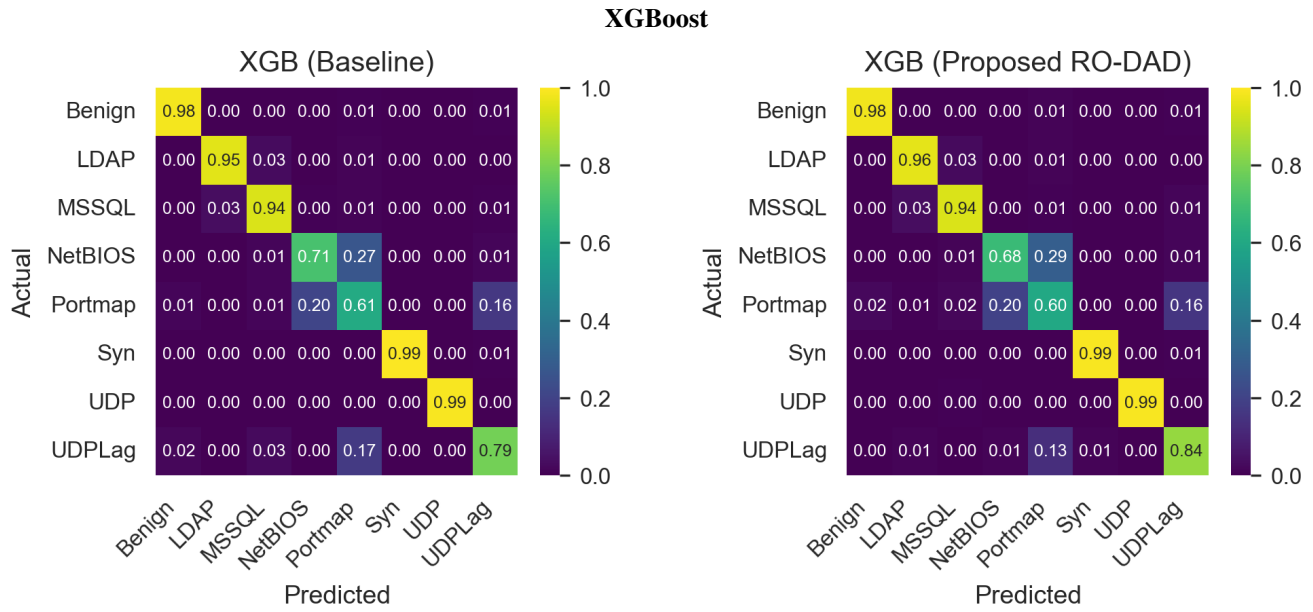


Figure 14. Confusion matrix comparison for XGBoost under baseline and RO-DAD framework.

existing studies while simultaneously reducing computational complexity through feature optimization.

Additionally, studies utilizing ensemble methods have shown high performance, although our study achieves competitive or comparable results. The integration of advanced models like Gradient Boosting and SVM⁴⁸ in other research provides competitive results. Still, our framework demonstrates consistent performance across evaluated metrics within the proposed experimental setting in handling DDoS detection.

Unlike most existing studies that primarily focus on maximizing classification performance, the proposed RO-DAD framework jointly optimizes detection effectiveness and computational efficiency through feature optimization. Experimental results demonstrate consistent reductions in training time, prediction time, and memory consumption, while maintaining competitive classification performance. It is important to note that differences in datasets, preprocessing steps, and evaluation protocols across studies limit strict direct comparison; therefore, the results should be interpreted as indicative rather than strictly equivalent.

Conclusions

In this study, we proposed the Resource-Optimized DDoS Attack Detection (RO-DAD) framework for effective and computationally efficient detection of DDoS attacks using the CICDDoS2019 dataset. The primary objective of this work was not only to achieve high detection performance, but also to reduce computational overhead in terms of training time, inference time, and memory consumption, making the approach suitable for cost-sensitive cloud environments. The proposed framework addresses key challenges in intrusion detection, including severe class imbalance and high dimensional feature spaces. A systematic preprocessing pipeline was applied, including data cleaning, stratified train-test splitting, feature scaling, and imbalance handling using RUS and SMOTE. To ensure robustness across different distribution scenarios, three distinct class balancing strategies (minimum, median, and mean thresholds) were evaluated. A three-stage feature optimization strategy was introduced, consisting of (i) domain knowledge based filtering to remove environment dependent features, (ii) statistical filtering to eliminate zero variance features, and (iii) model driven feature selection using tree-based importance ranking. This hybrid strategy significantly reduced feature dimensionality while preserving discriminative information, thereby improving both computational efficiency and classification performance. Extensive experiments were conducted using eight ML models, including both classical and ensemble methods. The results consistently demonstrate that the proposed RO-DAD framework maintains or improves detection performance across all evaluated configurations, while substantially reducing training time, inference time, and memory consumption. In particular, tree-based ensemble methods such as RF and XGBoost achieved the most stable performance across all dataset variations. To further strengthen the evaluation, confusion matrix analysis, class-wise performance assessment, and false positive rate analysis were performed. These results demonstrate improved minority class detection and reduced misclassification, without increasing false alarm rates. However, formal statistical significance testing was not conducted, which remains a limitation of this study.

Future work will incorporate statistical validation methods such as McNemar's test and Wilcoxon signed-rank tests to further confirm the robustness of the observed improvements. Although the framework was evaluated using a single benchmark dataset (CICDDoS2019), multiple internal dataset configurations were used to simulate varying class distributions, improving the reliability of the results. Nevertheless, external validation using real-world network traffic remains an important direction for future research to further evaluate generalization capability. In conclusion, the RO-DAD framework demonstrates that careful feature optimization can achieve a strong balance between detection accuracy and computational efficiency, making it a practical solution for real-world DDoS detection systems in cloud and network security environments.

Data Availability

The CICDDoS2019 dataset analyzed in this study is publicly available from the Canadian Institute for Cybersecurity (CIC) at <https://www.unb.ca/cic/datasets/ddos-2019.html>. Any additional processed data or materials generated during the current study are available from the corresponding author upon reasonable request.

References

1. Hassan, M. *et al.* Gitm: A gini index-based trust mechanism to mitigate and isolate sybil attack in rpl-enabled smart grid advanced metering infrastructures. *IEEE Access* **11**, 62697–62720 (2023).
2. Saheed, Y. K., Abiodun, A. I., Misra, S., Holone, M. K. & Colomo-Palacios, R. A machine learning-based intrusion detection for detecting internet of things network attacks. *Alex. Eng. J.* **61**, 9395–9409 (2022).
3. Imran, M., Durad, M. H., Khan, F. A. & Derhab, A. Reducing the effects of dos attacks in software defined networks using parallel flow installation. *Human-centric Comput. Inf. Sci.* **9**, 1–19 (2019).
4. Srinivasan, K., Mubarakali, A., Alqahtani, A. S. & Dinesh Kumar, A. A survey on the impact of ddos attacks in cloud computing: prevention, detection and mitigation techniques. In *Intelligent communication technologies and virtual mobile networks*, 252–270 (Springer, 2019).
5. Imran, M., Durad, M. H., Khan, F. A. & Abbas, H. Daisy: A detection and mitigation system against denial-of-service attacks in software-defined networks. *IEEE Syst. J.* **14**, 1933–1944 (2019).
6. Derhab, A., Belaoued, M., Guerroumi, M. & Khan, F. A. Two-factor mutual authentication offloading for mobile cloud computing. *IEEE Access* **8**, 28956–28969 (2020).
7. Ali, M. *et al.* Effective network intrusion detection using stacking-based ensemble approach. *Int. J. Inf. Secur.* **22**, 1781–1798 (2023).
8. Devi E. M, R., Almakayeel, N. & Lydia, E. L. Improved sand cat swarm optimization with deep learning based enhanced malicious activity recognition for cybersecurity. *Alex. Eng. J.* **98**, 187–198, DOI: <https://doi.org/10.1016/j.aej.2024.04.053> (2024).
9. Ahmad, Z., Khan, A. S., Shiang, C. W., Abdullah, J. & Ahmad, F. Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Transactions on Emerg. Telecommun. Technol.* **32**, e4150, DOI: [10.1002/ETT.4150](https://doi.org/10.1002/ETT.4150) (2021).
10. Sheng, S. & Wang, X. Network traffic anomaly detection method based on chaotic neural network. *Alex. Eng. J.* **77**, 567–579, DOI: <https://doi.org/10.1016/j.aej.2023.07.019> (2023).
11. Harini, R., Maheswari, N., Ganapathy, S. & Sivagami, M. An effective technique for detecting minority attacks in nids using deep learning and sampling approach. *Alex. Eng. J.* **78**, 469–482, DOI: <https://doi.org/10.1016/j.aej.2023.07.063> (2023).
12. Emirmahmutoğlu, E. & Atay, Y. A feature selection-driven machine learning framework for anomaly-based intrusion detection systems. *Peer-to-Peer Netw. Appl.* **18**, 161, DOI: [10.1007/s12083-025-01947-4](https://doi.org/10.1007/s12083-025-01947-4) (2025).
13. Breiman, L. Random forests. *Mach. Learn.* **45**, 5–32, DOI: [10.1023/A:1010933404324/METRICS](https://doi.org/10.1023/A:1010933404324/METRICS) (2001).
14. Najar, A. A. & Naik, S. M. AE-CIAM: A hybrid ai-enabled framework for low-rate DDoS attack detection in cloud computing. *Clust. Comput.* **28**, 103 (2025).
15. Sharafaldin, I., Lashkari, A. H., Hakak, S. & Ghorbani, A. A. Developing realistic distributed denial of service (ddos) attack dataset and taxonomy. In *2019 international carnaham conference on security technology (ICCSST)*, 1–8 (IEEE, 2019).
16. Ali, H. & Khan, F. A. Attributed multi-objective comprehensive learning particle swarm optimization for optimal security of networks. *Appl. soft computing* **13**, 3903–3921 (2013).

17. Ahmed, S., Tariq, N., Khan, F. A. & Chauhdary, S. H. Post-quantum dag-based blockchain for securing vehicular sensor networks. *IEEE Transactions on Consumer Electron.* (2026).
18. Rajagopal, S., Kundapur, P. P. & Hareesha, K. S. Towards effective network intrusion detection: From concept to creation on azure cloud. *IEEE Access* **9**, 19723–19742, DOI: [10.1109/ACCESS.2021.3054688](https://doi.org/10.1109/ACCESS.2021.3054688) (2021).
19. Thorat, O., Parekh, N. & Mangrulkar, R. Taxodacml: Taxonomy based divide and conquer using machine learning approach for ddos attack classification. *Int. J. Inf. Manag. Data Insights* **1**, 100048, DOI: [10.1016/J.JJIMEI.2021.100048](https://doi.org/10.1016/J.JJIMEI.2021.100048) (2021).
20. Ali, T. E., Chong, Y. W. & Manickam, S. Comparison of ml/dl approaches for detecting ddos attacks in sdn. *Appl. Sci.* **2023**, Vol. 13, Page 3033 **13**, 3033, DOI: [10.3390/APP13053033](https://doi.org/10.3390/APP13053033) (2023).
21. Alzahrani, A. O. & Alenazi, M. J. Designing a network intrusion detection system based on machine learning for software defined networks. *Futur. Internet* **13**, DOI: [10.3390/fi13050111](https://doi.org/10.3390/fi13050111) (2021).
22. Virupakshar, K. B. *et al.* Distributed denial of service (ddos) attacks detection system for openstack-based private cloud. *Procedia Comput. Sci.* **167**, 2297–2307, DOI: [10.1016/J.PROCS.2020.03.282](https://doi.org/10.1016/J.PROCS.2020.03.282) (2020).
23. Gumaste, S., Narayan, D. G., Shinde, S. & Amit, K. Detection of ddos attacks in openstack-based private cloud using apache spark. *J. Telecommun. Inf. Technol.* **2020**, 62–71, DOI: [10.26636/JTIT.2020.146120](https://doi.org/10.26636/JTIT.2020.146120) (2020).
24. Dhanapal, A. & Nithyanandam, P. An openstack based cloud testbed framework for evaluating http flooding attacks. *Wirel. Networks* **27**, 5491–5501, DOI: [10.1007/s11276-019-01937-4](https://doi.org/10.1007/s11276-019-01937-4) (2021).
25. ul haque, M., Asghar, H., Durad, M. H. & Usman, A. Detection of internet layer dos attack on cloud using deep learning. In *2023 20th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*, 600–605, DOI: [10.1109/IBCAST59916.2023.10713027](https://doi.org/10.1109/IBCAST59916.2023.10713027) (2023).
26. Kushwah, G. S. & Ranga, V. Detecting ddos attacks in cloud computing using extreme learning machine and adaptive differential evolution. *Wirel. Pers. Commun.* **124**, 2613–2636, DOI: [10.1007/s11277-022-09481-9](https://doi.org/10.1007/s11277-022-09481-9) (2022).
27. Kushwah, G. S. & Ranga, V. Voting extreme learning machine based distributed denial of service attack detection in cloud computing. *J. Inf. Secur. Appl.* **53**, 102532, DOI: [10.1016/J.JISA.2020.102532](https://doi.org/10.1016/J.JISA.2020.102532) (2020).
28. Alqarni, A. A. Majority vote-based ensemble approach for distributed denial of service attack detection in cloud computing. *J. Cyber Secur. Mobil.* **11**, 265–278, DOI: [10.13052/JCSM2245-1439.1126](https://doi.org/10.13052/JCSM2245-1439.1126) (2022).
29. Mugunthan, S. Soft computing based autonomous low rate ddos attack detection and security for cloud computing. *J. Soft Comput. Paradig.(JSCP)* **1**, 80–90 (2019).
30. Bahashwan, A. A. *et al.* A systematic literature review on machine learning and deep learning approaches for detecting ddos attacks in software-defined networking. *Sensors* **2023**, Vol. 23, Page 4441 **23**, 4441, DOI: [10.3390/S23094441](https://doi.org/10.3390/S23094441) (2023).
31. Aljuhani, A. Machine learning approaches for combating distributed denial of service attacks in modern networking environments. *IEEE Access* **9**, 42236–42264, DOI: [10.1109/ACCESS.2021.3062909](https://doi.org/10.1109/ACCESS.2021.3062909) (2021).
32. Ullah, Z., Al-Turjman, F., Mostarda, L. & Gagliardi, R. Applications of artificial intelligence and machine learning in smart cities. *Comput. Commun.* **154**, 313–323 (2020).
33. Krones, F., Marikkar, U., Parsons, G., Szmul, A. & Mahdi, A. Review of multimodal machine learning approaches in healthcare. *Inf. Fusion* **114**, 102690, DOI: <https://doi.org/10.1016/j.inffus.2024.102690> (2025).
34. Al-Garadi, M. A. *et al.* A survey of machine and deep learning methods for internet of things (iot) security. *IEEE Commun. Surv. Tutorials* **22**, 1646–1685, DOI: [10.1109/COMST.2020.2988293](https://doi.org/10.1109/COMST.2020.2988293) (2020).
35. Kilincer, I. F., Ertam, F. & Sengur, A. Machine learning methods for cyber security intrusion detection: Datasets and comparative study. *Comput. Networks* **188**, 107840, DOI: [10.1016/J.COMNET.2021.107840](https://doi.org/10.1016/J.COMNET.2021.107840) (2021).
36. Pandey, N. & Mishra, P. K. Detection of ddos attack in iot traffic using ensemble machine learning techniques. *Networks Heterog. Media* **18**, 1393–1409, DOI: [10.3934/nhm.2023061](https://doi.org/10.3934/nhm.2023061) (2023).
37. Wilson, H. A systematic review of machine learning approaches for intrusion detection: Insights from logistic regression. *SSRN Prepr.* (2025). Highlights logistic regression's strengths in computational efficiency and transparency.
38. Liu, H. & Lang, B. Machine learning and deep learning methods for intrusion detection systems: A survey. *Appl. Sci.* **2019**, Vol. 9, Page 4396 **9**, 4396, DOI: [10.3390/APP9204396](https://doi.org/10.3390/APP9204396) (2019).
39. Moulahi, T., Zidi, S., Alabdulatif, A. & Atiquzzaman, M. Comparative performance evaluation of intrusion detection based on machine learning in in-vehicle controller area network bus. *IEEE Access* **9**, 99595–99605, DOI: [10.1109/ACCESS.2021.3095962](https://doi.org/10.1109/ACCESS.2021.3095962) (2021).

40. Ünal Çavuşoğlu. A new hybrid approach for intrusion detection using machine learning methods. *Appl. Intell.* **49**, 2735–2761, DOI: [10.1007/S10489-018-01408-X/TABLES/16](https://doi.org/10.1007/S10489-018-01408-X/TABLES/16) (2019).
41. Chawla, N. V., Bowyer, K. W., Hall, L. O. & Kegelmeyer, W. P. Smote: Synthetic minority over-sampling technique. *J. Artif. Intell. Res.* **16**, 321–357, DOI: [10.1613/jair.953](https://doi.org/10.1613/jair.953) (2002).
42. Fernández, A., Garcia, S., Herrera, F. & Chawla, N. V. Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary. *J. artificial intelligence research* **61**, 863–905 (2018).
43. Alfatemi, A., Rahouti, M., Amin, R. *et al.* Advancing ddos attack detection: A synergistic approach using deep residual neural networks and synthetic oversampling. *arXiv preprint* (2024).
44. Harahsheh, K., Al-Naimat, R. & Chen, C. Using feature selection enhancement to evaluate attack detection in the internet of things environment. *Electronics* **13**, 1678, DOI: [10.3390/electronics13091678](https://doi.org/10.3390/electronics13091678) (2024).
45. Ayad, A., Sakr, N. & Hikal, N. A hybrid approach for efficient feature selection in anomaly intrusion detection for iot networks. *The J. Supercomput.* **80**, 26942–26984, DOI: [10.1007/s11227-024-06409-x](https://doi.org/10.1007/s11227-024-06409-x) (2024).
46. Kumar, R. & Singh, S. P. Efficient feature reduction techniques for anomaly detection in software-defined networks. *The J. Supercomput.* DOI: [10.1007/s11227-024-06409-x](https://doi.org/10.1007/s11227-024-06409-x) (2024).
47. Rajathi, C. & Panjanathan, R. A two-phase feature selection framework for intrusion detection system: Balancing relevance and computational efficiency (2p-fsid). *Appl. Artif. Intell.* **39**, 2539396, DOI: [10.1080/08839514.2025.2539396](https://doi.org/10.1080/08839514.2025.2539396) (2025). <https://doi.org/10.1080/08839514.2025.2539396>.
48. Sakr, H. A. *et al.* Machine learning-based detection of ddos attacks on iot devices in multi-energy systems. *Egypt. Informatics J.* **28**, 100540, DOI: [10.1016/J.EIJ.2024.100540](https://doi.org/10.1016/J.EIJ.2024.100540) (2024).
49. Kumar, D., Pateriya, R. K., Gupta, R. K., Dehalwar, V. & Sharma, A. Ddos detection using deep learning. *Procedia Comput. Sci.* **218**, 2420–2429, DOI: [10.1016/J.PROCS.2023.01.217](https://doi.org/10.1016/J.PROCS.2023.01.217) (2023).
50. Anley, M. B., Genovese, A., Agostinello, D. & Piuri, V. Robust ddos attack detection with adaptive transfer learning. *Comput. & Secur.* **144**, 103962, DOI: [10.1016/J.COSE.2024.103962](https://doi.org/10.1016/J.COSE.2024.103962) (2024).

Acknowledgements

The authors express their gratitude to King Saud University, Riyadh, Saudi Arabia, for its support through the Ongoing Research Funding Program (ORF-RICSP-2026-4) and acknowledge the assistance from the Critical Infrastructure Protection and Malware Analysis (CIPMA) Lab at PIEAS.

Funding

This research was supported by Ongoing Research Funding Program (ORF-RICSP-2026-4), King Saud University, Riyadh, Saudi Arabia. It was also supported by the National Center for Cyber Security (NCCS), Pakistan, under the Higher Education Commission (HEC) of Pakistan, through the Critical Infrastructure Protection and Malware Analysis (CIPMA) Lab at the PIEAS.

Author contributions

M.U.H. contributed to the conceptualization, methodology, software, data curation, formal analysis, investigation, visualization, and writing—original draft. M.H.D. and A.U. supervised the work, contributed to project administration, and writing—review and editing. F.A.K. contributed to funding acquisition, validation, and writing—review and editing. S.A. contributed to validation, resources, and writing—review and editing. All authors reviewed and approved the final manuscript.

Declarations

Competing interests

The authors declare no competing interests.

Table 14. Performance Comparison of the Proposed Technique with Existing Techniques in the Literature

Study	Year	ML Model	Accuracy	Precision	Recall	F1-Score
Sharafaldin et al. ¹⁵	2019	ID3	NA	0.780	0.650	0.690
		RF	NA	0.770	0.560	0.620
		NB	NA	0.410	0.110	0.050
		LR	NA	0.250	0.020	0.040
Alzahrani and Alenazi ²¹	2021	KNN	0.980	0.990	0.990	0.990
		SVM	0.860	0.860	0.870	0.850
		NB	0.450	0.660	0.540	0.380
		DT	0.990	0.990	0.990	0.990
		RF	0.990	0.990	0.990	0.990
		LR	0.980	0.990	0.980	0.990
Alqarni ²⁸	2022	NB	0.962	NA	0.923	NA
		DT	0.953	NA	0.927	NA
		SVM	0.959	NA	0.930	NA
		KNN	0.960	NA	0.922	NA
		Ensemble	0.980	NA	0.975	NA
Ali et al. ²⁰	2023	SVM	0.956	NA	NA	0.970
		KNN	0.906	NA	NA	0.880
		DT	0.953	NA	NA	0.960
		MLP	0.911	NA	NA	0.890
		CNN	0.901	NA	NA	0.840
Kumar et al. ⁴⁹	2023	LSTM	0.980	0.980	0.970	0.970
Anley et al. ⁵⁰	2024	VGG16	0.922	0.923	0.926	0.923
		VGG19	0.927	0.922	0.923	0.929
		ResNet50	0.917	0.918	0.912	0.910
Sakr et al. ⁴⁸	2024	DT	0.984	0.370	0.402	0.381
		GB	0.989	0.426	0.409	0.416
		SVM	0.984	0.315	0.244	0.262
		KNN	0.988	0.508	0.417	0.445
		RF	0.987	0.379	0.435	0.393
		GB+SVM	0.988	0.383	0.416	0.398
This Study	2026	DT	0.989	0.994	0.989	0.991
		RF	0.990	0.994	0.990	0.992
		AdaBoost	0.975	0.973	0.975	0.973
		XGBoost	0.982	0.991	0.982	0.986
		LR	0.954	0.989	0.954	0.971
		GNB	0.950	0.985	0.950	0.963