FULafia Journal of Science & Technology
ISSN (print) 2449-0954 \ ISSN (online) 2636-4972
https://lafiascijournals.org.ng/index.php/fjst
Sponsored By TETFUND TERTIARY EDUCATION TRUST FUND
Manuscript => Received: Dec., 2025; Accepted: Feb., 2026; Published: March., 2026 https://doi.org/10.62050/fjst2026.v10n1.685

AN INTELLIGENT FEATURE-OPTIMIZED FRAMEWORK FOR DDoS ATTACK DETECTION USING SVM and XGBOOST

Moshood A. Hambali [1]*, Beatrice A. Ayati[2], Yinusa A. Olasupo[1], Mu'awuya Dalhatu[2] and Adesina A. Akinyemi[2]
[1]*Department of Software Engineering, Federal University Wukari, PMB 1020, Nigeria*
[2]*Department of Computer Science, Federal University Wukari, PMB 1020, Nigeria*
***Corresponding email: hambali@fuwukari.edu.ng

ABSTRACT
Distributed Denial of Service (DDoS) attacks remain a serious issue for network security, especially within cloud systems that rely on shared and scalable resources. In such settings, attackers can easily use large groups of compromised devices to flood targets with traffic, making genuine access almost impossible. Traditional intrusion detection systems (IDS) that rely on fixed signatures or manually defined rules often fail to keep up, since new and fast-changing attacks rarely fit known patterns. To improve detection, this research applied a machine learning–based approach that combines Support Vector Machine (SVM) and Extreme Gradient Boosting (XGBoost) models, supported by the Fisher Score Algorithm (FSA) for feature selection. The well-known CIC-DDoS2019 dataset from Kaggle was used because it includes both modern DDoS attacks and regular network traffic, giving a practical base for testing the models. Our workflow followed three stages: first, cleaning and preparing the data; second, selecting key features using FSA; and finally, training and evaluating the classifiers. We measured performance with accuracy, precision, recall, F1-score, and the Area Under the Curve (AUC). XGBoost achieved the highest results of 97 % accuracy and an AUC of 0.99, while SVM's accuracy fell to 65 % after feature reduction, despite a small AUC gain to 0.65. The results of this study demonstrate that combining machine learning models of XGBoost with FSA significantly improves the detection of DDoS attacks in cloud environments. This highlights its potential for superior performance over traditional IDS methods.

Keywords: Distributed denial of service; Network intrusion detection systems; Machine learning; XGBoost algorithm; Fisher score algorithm

INTRODUCTION
In recent years, Distributed Denial of Service (DDoS) attacks have become one of the biggest headaches for people working in network security. The problem is even worse in cloud environments, where many users share the same servers and resources. In these settings, it doesn't take much for a burst of harmful traffic to slow things down or completely knock services offline. Most large-scale DDoS incidents involve thousands of infected devices working together to send data toward one target, overwhelming it until normal access becomes impossible (Sambangi & Gondi, 2020).

A regular Denial of Service (DoS) attack follows the same basic idea; it overloads a system with too many requests, but usually comes from one computer or source. As a result of that, it's easier to identify and stop (Iraq *et al*., 2025). When attackers use many devices at once, however, it becomes a different story. These devices, often part of what's known as a botnet, work together to produce enormous waves of fake traffic that can take down even strong network systems (Mahboubi *et al*., 2025).

DDoS attacks can hit different layers of a network depending on the method being used. Some focus on the network or transport layer, while others target the application layer (Hadeel & Esamaddin, 2020). For example, fragmentation attacks send huge amounts of broken data packets that use up processing power (Berhanu & Assamnew, 2024). Bandwidth attacks fill up the network with junk traffic until it can't handle any more, and application-level attacks often take advantage of flaws in specific software to disrupt services (Xu *et al*., 2024).

As technology grows and networks become more complex, defending against DDoS attacks has turned into a constant race. Traditional intrusion detection systems (IDS) based on static rules or stored signatures can't always keep pace with new attack strategies (Asaju *et al*., 2017; Khan *et al*., 2025). Rule-based systems, for instance, need constant human updating, while signature-based ones can only detect what they already know (Xie *et al*., 2023). This delay gives attackers a dangerous advantage. Due to these issues, many researchers now look toward machine learning (ML) approaches to improve detection. ML models can recognize patterns in data and adjust to new situations without as much manual effort (Hambali & Oforjetu, 2024; Berbiche & El Alami, 2024).

Recent research has highlighted persistent challenges faced by machine-learning (ML) models when

processing datasets containing a high number of variables (Raza *et al*., 2024). When data are excessively detailed or multidimensional, models may analyse irrelevant features, which increases computational cost and reduces accuracy. Feature selection provides an effective means of addressing this problem by identifying the most informative attributes and discarding redundant ones. Within cybersecurity applications, this process is essential for distinguishing DDoS traffic from legitimate network activity.

To mitigate issues related to high-dimensional data, this study employs the Fisher Score Algorithm (FSA). The FSA is a supervised, filter-based feature-selection technique that assigns a discriminative score to each feature according to its relevance to the classification task (Li *et al*., 2024). The approach offers advantages in both speed and computational efficiency while maintaining classification accuracy, making it particularly suitable for large-scale or complex network datasets. DDoS attack strategies have evolved to become increasingly sophisticated and unpredictable, reducing the effectiveness of traditional signature-based detection systems. Because these systems depend on predefined attack signatures, they are unable to identify novel or modified attack patterns until manual updates are applied. This delay enables adversaries to exploit vulnerabilities before defenses are refreshed. Recent studies indicate that DDoS incidents are rising in both frequency and complexity, frequently overwhelming network resources and preventing legitimate users from accessing online services (Xu *et al*., 2024).

Conventional rule-based security mechanisms are similarly limited, as they require continuous configuration and maintenance. These constraints have accelerated the shift toward adaptive, intelligent detection systems driven by artificial intelligence. Accordingly, this work proposes a hybrid detection framework that integrates SVM and Extreme Gradient Boosting (XGBoost) algorithms with Fisher-Score-based feature reduction. The objective is to enhance detection accuracy, computational efficiency, and adaptability in mitigating DDoS attacks within modern network-security environments. The rest of this study is divided into the following sections: Related works, Research method, Results and Discussion, and Conclusion.

The rapid expansion of the internet and the growing dependence on digital infrastructure have intensified cybersecurity challenges, especially those linked to DDoS attacks (AlSaleh *et al*., 2024). As digital platforms become vital to both individuals and organizations, the frequency and complexity of attacks targeting network availability have risen significantly. DDoS attacks typically overwhelm network resources, services, or applications by directing excessive traffic from multiple compromised devices, collectively known as botnets (Uddin *et al*., 2024). These botnets can launch extensive assaults by distributing malicious software across numerous devices, which then generate massive amounts of Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) packets, severely disrupting service availability (Pakmehr *et al*., 2024).

DoS and DDoS attacks share the same objective, that is, disrupting online services through excessive traffic (Setitra *et al*., 2024). A DoS attack usually originates from a single source, exhausting the target system's capacity through techniques such as TCP SYN floods, Ping of Death, or packet fragmentation attacks (Alhasawi & Alghamdi, 2024). Although disruptive, DoS attacks are relatively easier to detect and mitigate because of their single-point origin.

Conversely, DDoS attacks are more complex, leveraging multiple compromised systems to launch large-scale, coordinated assaults. These can be categorized into three primary types (Brown *et al*., 2024):

i) **Application layer attacks:** These attacks target specific applications or services to exhaust server resources, often employing techniques such as Hypertext Transfer Protocol (HTTP) floods.
ii) **Bandwidth/volume attacks:** These attacks focus on saturating network bandwidth with high traffic volumes, rendering services inaccessible.
iii) **Traffic/fragmentation attacks:** These attacks use fragmented packets to overwhelm the target, forcing it to expend resources on reassembling traffic.

Figure 1 illustrates the concept of a DDoS attack**,** where multiple compromised devices simultaneously send large volumes of traffic toward a target server or network. The excessive traffic overwhelms the target's resources, causing slow performance or complete service unavailability for legitimate users.

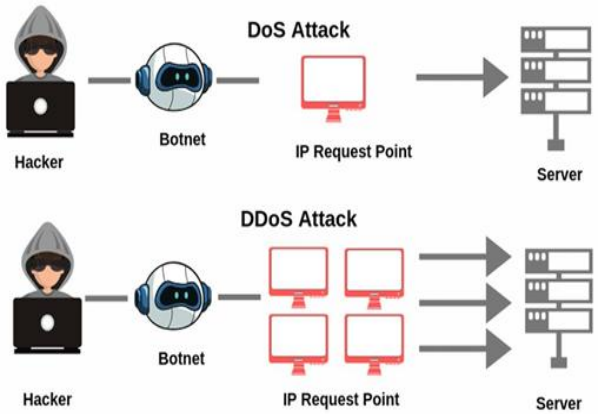


Figure 1: DDOS attack (Merkebaiuly, 2024)

The impact of DDoS attacks extends significantly into cloud computing environments, leveraging the "pay-as-you-go" model to achieve their objectives. Cloud computing has gained tremendous popularity due to its flexibility and cost-effectiveness. However, this popularity has also made cloud services a prime target for DDoS attacks (Songa & Karri, 2024). Key features of cloud computing that contribute to its vulnerability include:

i) **Resource utilization:** DDoS attackers exploit cloud computing's resource allocation features, such as Central Processing Unit (CPU), network bandwidth, and storage space, to overwhelm target servers.
ii) **Configuration and node disruption:** Attackers may alter or destroy configuration information and critical nodes within the physical network.
iii) **Service access:** Unauthorized access to cloud services through programming and other means can further exacerbate the impact of DDoS attacks.

To combat these sophisticated threats, machine learning (ML) and deep learning (DL) techniques have emerged as powerful tools for DDoS detection and prevention. ML and DL models offer several key advantages in this context:

i) **Traffic analysis:** ML algorithms can analyze network traffic patterns to detect anomalies indicative of DDoS attacks. By examining traffic characteristics, these models can identify deviations from normal behavior that may signal an ongoing attack.
ii) **Pattern recognition:** Leveraging historical attack data, ML models can recognize specific characteristics and patterns associated with various DDoS attack types. This learning enables models to differentiate between legitimate and malicious traffic more effectively.
iii) **Real-time detection:** ML models provide near real-time detection of DDoS attacks by continuously monitoring network traffic and flagging suspicious activities based on learned patterns. This capability is crucial for mitigating attacks before they cause significant disruption.

The growing complexity and frequency of DDoS attacks have prompted significant research into effective detection and mitigation strategies, particularly in the context of Internet of a Thing (IoT) networks. Various approaches have been explored to counter these attacks, leveraging both machine learning and block-chain technologies.

Ajagbe *et al*. (2024) proposed an IDS structured around four key stages: data aggregation, preprocessing, model training, and attack detection. Data aggregation organizes network packets, while preprocessing selects critical features to distinguish between normal and malicious activities, similar to the Knowledge Discovery in Databases (KDD) Cup 99 dataset. The study introduced a novel Flexible Mutual Information Feature Selection (FMIFS) algorithm, adapted from Battiti's approach, to minimize redundant features and improve model efficiency. Combined with a Least Squares Support Vector Machine (LSSVM) classifier, the system simplifies complex optimization tasks and enhances computational performance. Evaluated using the KDD Cup 99, NSL-KDD, and Kyoto 2006+ datasets, the proposed FMIFS-LSSVM IDS achieved a 95.8 % accuracy rate with improved efficiency in detecting intrusions.

Jawahar *et al*.(2024) introduced a hybrid defense framework that integrates predictive analytics with decentralized storage, utilizing the Ethereum block-chain to maintain a secure blacklist of malicious IP addresses. Using the CICDDoS2019 dataset, several classifiers were trained, with the Artificial Neural Network (ANN) achieving the highest accuracy and real-time detection performance of 72.49 % in a Mininet virtual environment. Similarly, (Pang *et al*., 2023) developed the Pairwise Relation Prediction Network (PReNet), a deep weakly supervised anomaly detection model that learns relational patterns between data pairs to identify both known and novel anomalies. Santos-Neto *et al*. (2024) proposed ML-Entropy, a hybrid approach combining entropy-based metrics with machine learning to dynamically adjust thresholds, improving accuracy and convergence while achieving over 99% detection accuracy on the DARPA and enterprise datasets. In the context of vehicular networks, Setia *et al*. (2024) designed a machine learning–based architecture for VANET Cloud environments that achieved 99.59 % accuracy in detecting and mitigating DDoS attacks, thereby enhancing real-time threat response and system reliability. Altulaihan *et al*. (2024) examined multiple classifiers for IoT-based DDoS detection, finding that Decision Tree and Random Forest models optimized through Genetic Algorithm–based feature selection reached perfect accuracy and efficiency, outperforming SVMs on large and correlated datasets. Lastly, Patel *et al*. (2024) applied machine learning techniques within Next-Generation Firewalls, where the Binary Decision Tree achieved 99 % accuracy and superior prediction speed, significantly improving real-time DDoS detection and network resilience.

Summary of Related Works

Table 1 provides a summary of the recent related works in the domain of DDoS.

Table 1: Summary of related works

Author(s) & Year	Title/Focus	Key Methods	Datasets	Main Findings
Pang *et al.* (2023)	Pairwise Relation Prediction Network (PReNet) for Weakly Supervised Anomaly Detection	Deep Learning; Pairwise Relation Prediction; Weak Supervision	Benchmark anomaly detection datasets	Detected both known and novel anomalies through relational feature learning, improving generalization.
Santos-Neto *et al.* (2024)	ML-Entropy: Hybrid Entropy and Machine Learning-Based DDoS Detection	Entropy-based statistical analysis; Random Forest; Support Vector Classifier	DARPA dataset; real enterprise traffic	ML-Entropy achieved >99 % accuracy; dynamically adjusted entropy thresholds improved convergence and reduced false alarms.
Setia *et al.* (2024)	Machine Learning-Driven DDoS Attack Detection in VANET Cloud Environments	Random Forest, SVM, ANN; VANET security framework	Custom VANET Cloud dataset	Achieved 99.59 % precision; proposed adaptive ML architecture enhanced DDoS resilience in connected vehicles.
Altulaihan *et al.* (2024)	Classifier-Based DDoS Detection in IoT Systems Using Feature Selection	Decision Tree, Random Forest, KNN, SVM; Genetic Algorithm (GA); Correlation-based Feature Selection (CFS)	IoT traffic dataset	DT and RF with GA features achieved 100 % accuracy; SVM underperformed with CFS; emphasized importance of optimal feature selection.
Patel *et al.* (2024)	Machine Learning Model for DDoS Detection in Next Generation Firewall (NGF) Systems	Binary Decision Tree, XGBoost, SVM	Real network traffic (NGF dataset)	Binary Decision Tree achieved 99 % accuracy and fastest prediction; improved NGF filtering efficiency for DDoS attacks.
Ajagbe *et al.* (2024)	Network Intrusion Detection on Distributed Denial of Service (DDoS) Attacks	SVM, XGBoost, Fisher Score Algorithm (FSA)	CIC-DDoS2019 (Kaggle)	XGBoost with FSA achieved 97 % accuracy and AUC = 0.99; demonstrated scalability and adaptability in real-time detection.
Communication System Engineering, Baghdad, Iraq *et al.* (2025)	Analysis of Denial of Service (DoS) Attack Mechanisms	Network traffic simulation; single-source DoS characterization	Simulated DoS traffic	Highlighted detection simplicity of single-source DoS attacks; served as baseline for understanding DDoS evolution.
Mahboubi *et al.* (2025)	Botnet-Based Distributed Denial of Service (DDoS) Attacks: A Comprehensive Analysis	Botnet traffic modeling; distributed network simulation	Synthetic DDoS dataset	Demonstrated how botnet-coordinated attacks amplify traffic intensity and complexity, overwhelming traditional defense systems.
Khan *et al.* (2025)	Limitations of Traditional Intrusion Detection Systems in DDoS Mitigation	Signature-based and rule-based detection analysis	Benchmark IDS datasets	Found that static rule-based systems fail to adapt to evolving DDoS patterns; underscored need for ML-driven adaptive models.

Current research on DDoS detection using machine learning lacks comprehensive studies integrating SVM and XGBoost within a unified framework. Exploring this combination could lead to a more adaptive and efficient detection model with improved accuracy, reduced false positives, and enhanced real-time performance.

MATERIALS AND METHODS

Conceptual Framework

The research methodology is divided into three principal phases: data preprocessing, feature selection, and classification and performance evaluation. The initial phase, data preprocessing, involves preparing and refining raw data for effective analysis. This includes handling missing values, normalizing data, and eliminating noise or outliers to ensure the data is suitable for further analysis.

The next phase, feature selection, focuses on identifying and extracting key attributes from the preprocessed data. The goal is to retain only those features that significantly influence the detection of network intrusions. The FSA algorithm is proposed for this phase, as it reduces dimensionality, enhances model performance, and improves result interpretability.

In the final phase, classification and performance evaluation, the features selected through the FSA algorithm are used to train machine learning classification models using the SVM and XGBoost algorithm, with model rigorously evaluated using metrics like accuracy, precision, recall, and F1-score. This evaluation process ensures the models' reliability and efficiency in detecting DDOS network intrusions.

Figure 2 provides a visual representation of the proposed methodology, offering a comprehensive overview of the sequential progression through the outlined phases. This framework not only addresses the complexities of network intrusion detection but also contributes valuable insights to the broader field of cybersecurity research.

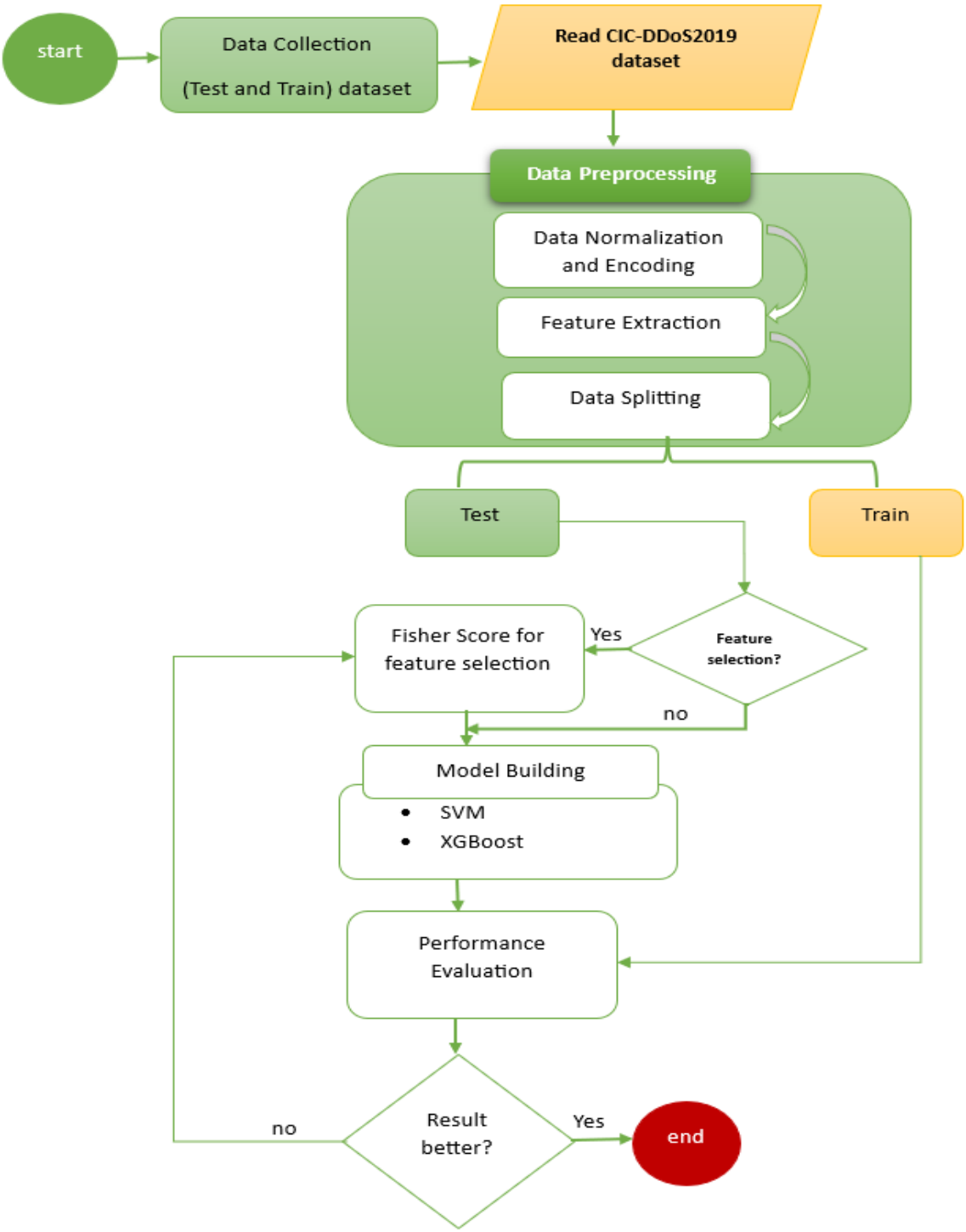


Figure 2: Research Methodology

Dataset Description

This study used the CIC-DDoS2019 dataset (Ramakrishnan *et al.*, 2025), which is given by the Canadian Institute for Cybersecurity and acquired from the Mendeley machine learning repository (https://data.mendeley.com/datasets/ssnc74xm6r/1), in order to detect DDOS attacks. CIC-DDoS2019 encompasses both benign and typical DDoS attacks, rendering it a useful reflection of actual network traffic (PCAPs). The dataset consists of labelled flows generated by CICFlowMeter-V3. These flows contain timestamps, source and destination IPs, ports, protocols, and attack kinds. The information is kept in Comma-Separated Values (CSV) files. Data was collected on a daily basis, with each day including raw network traffic (PCAPs) and event logs (Windows and Ubuntu event logs) for each system. The authors utilized CICFlowMeter-V3 to extract more than 80 traffic characteristics from the raw data, storing them as CSV files for each machine. The dataset includes a range of

contemporary reflected DDoS assaults, such as PortMap, NetBIOS, LDAP, MSSQL, UDP, UDP-Lag, SYN, NTP, DNS, and SNMP.

Data Preprocessing

Upon examination of the proposed dataset, it was noted that some features display nominal values. Therefore, it is essential to transform these features into numerical values throughout both the training and testing stages. Algorithm 1 outlines the process for converting nominal qualities into numerical values (Schema 1). At first, the algorithm computes the overall count of nominal characteristics. Afterwards, specific information is provided for each nominal feature, including the name of the feature, the number of variables (k), and the frequency of each variable inside the feature. Subsequently, the repeating variables are arranged in a descending order according to their frequency of recurrence, and each variable is subsequently assigned a numerical value ranging from zero to K.

Algorithm 1: Nominal to Numeric Algorithm

Step 1: nominal feature counter: c
Step 2: set $i = 1$
Step 3: while i ! =c then
Step 4: k_i: num of variable (i) involved in feature C; p_i: num of repeated variable (i) repeated
Step 5: order repeated variable from maximum to minimum number
Step 6: allocate the number from zero to k that was obtained from step 2 to variables
Step 7: $i = i + 1$ go to step 3
Step 1: stop

Schema 1: Nominal to numeric algorithm

Algorithm 2: Fisher Score Selection Approach

Input: f → feature, **S**→ features set, **C**→ classes, **N**→ number of samples in a class, $\boldsymbol{\mu}$→ mean values of features
For f in S:
  Compute **mean(f)**
  Compute **variance(f)**
Calculate **overall mean**
Calculate **Between-Class Variance using** $\sum_{k=1}^{c} \boldsymbol{N_i(\mu_i - \mu)^2}$
Calculate Within-Class Variance **using** $\sum_{k=1}^{c} \frac{1}{N_i - 1} \sum_{x \epsilon C^k} (x - \mu^k)^2$
Compute fisher score using equation (1)

Schema 2: Fisher score selection approach

Feature Selection

Feature selection is a technique used to reduce the number of features in the dataset by removing unnecessary ones, leaving only the most important ones intact. Feature selection is utilized in the field of DDoS detection to enhance the efficiency of the detection process. The objective is to identify and preserve the most relevant characteristics while disregarding those that do not significantly contribute to the task of detection. Therefore, in order to determine the important characteristics, the study recommended using an artificial intelligence method, namely the FSA algorithm.

The algorithm finds a feature subset in a data space spanned by the selected features, and maximizes the distances between data points in different classes while minimizing the distances between data points in the same class. For instance, in the adapted dataset, given a training instance $X \epsilon \mathbb{R}^{mxn}$ with respect to c different classes, the Fisher score of the ith feature is computed as:

$$FS(f_i) = \frac{S_b(f_i)}{\sum_{k=1}^{c} S_t^{(k)}(f_i)} \qquad 1$$

Where f is the features, k is an instance class at a point while c is the total classes, $S_b(f_i)$ is *Between-class variance* of feature f_i, $S_t^{(k)}(f_i)$ is *Within-class variance* (or total scatter) of feature f_i for class k.

Essentially, FSA is based on the idea of measuring the ratio of between-class variance to within-class variance for each feature. The higher the Fisher Score, the better the feature is at distinguishing between classes. The FSA algorithm is shown on algorithm 2 (Schema 2).

Classification Algorithm

The examination of DDOS network anomaly detection using the CIC-DDoS2019 dataset involves building and evaluating models using a supervised machine learning framework. Specifically, classification approaches, due to the dataset's classification problem of network attacks. The main goal is to forecast the class

designation of each sample using the characteristics found in the CIC-DDoS2019 dataset. Therefore, the study proposed the application of machine-learning techniques to build DDOS detection models, specifically the SVM and XGBoost algorithms.

SVM Algorithm

The SVM is a highly effective supervised machine learning technique that is commonly employed for classification purposes, such as identifying DDoS attacks. The SVM algorithm operates by mapping the input data into a space with a higher number of dimensions, facilitating the identification of distinct boundaries between various classes. The attainment of this transformation is accomplished by employing kernel functions.

The hyperplane in Support Vector Machines (SVM) refers to the decision boundary that optimally maximizes the distance between the two classes. The margin is the measure of the separation between the hyperplane and the nearest data points from each class, which are referred to as support vectors. The optimal hyperplane is the one that maximizes the margin, hence enhancing the ability to generalize to new and unknown data. The support vectors play a critical role in DDoS attack detection by establishing the boundary that distinguishes regular traffic from malicious traffic.

The SVM technique can effectively detect DDoS attacks by utilizing several kernels, allowing it to adjust to the inherent structure of the data. This adaptability makes SVM a versatile and resilient solution. SVM's capacity to analyze both linear and non-linear patterns in network traffic renders it a significant asset in the field of cybersecurity, successfully discerning between benign and dangerous activity within the network. The SVM algorithm is shown on algorithm 3 (Schema 3).

Algorithm 3: SVM Algorithm

Input:
Dataset (D):
Feature Set (F):
Hyperparameters (C, γ):
Training Data (T):
Testing Data (Test):
Output:
Prediction: Class labels for each sample in the test set (Normal or DDoS).
Steps:

1. **Load and Preprocess Data:**
   - Load the CIC-DDoS2019 dataset.
   - Preprocess the data (e.g., handle missing values, normalize/standardize features).
   - Split the dataset into training and testing sets (e.g., 70 % training, 30 % testing).
2. **Feature Selection (Fisher Score Algorithm):**
   - Apply the Fisher Score Algorithm (FSA) to rank the features.
   - Select the top relevant features based on Fisher score ranking.
3. **Initialize SVM Model:**
   - Choose an SVM kernel (typically Radial Basis Function (RBF)).
   - Set hyperparameters:
     - **C:** Regularization parameter (controls the trade-off between margin size and classification error).
     - **γ:** Kernel coefficient (determines the shape of the decision boundary).
4. **Train the SVM Model**:
5. **Make Predictions on Test Data**:
6. **Evaluate Model**:
   - Calculate the evaluation metrics to assess the model's performance:
     - **Accuracy:** Proportion of correct predictions.
     - **Precision:** Proportion of true positives among all predicted positives.
     - **Recall:** Proportion of true positives among all actual positives.
     - **F1-Score:** Harmonic mean of precision and recall.
     - **Area Under the Curve (AUC):** Evaluate the model's ability to distinguish between classes.
7. **Output Results**

Schema 3: SVM algorithm

Algorithm 4: XGBoost Algorithm

Pseudocode for XGBoost DDoS Detection Model

Input:

- **Dataset (D):** A dataset containing both DDoS attack data and regular network traffic.
- **Feature Set (F):** The set of features extracted from the dataset.
- **Training Data (T):** Subset of the dataset for training the model.
- **Testing Data (Test):** Subset of the dataset for evaluating the model.

Output:

- **Prediction:** Class labels for each sample in the test set (Normal or DDoS).

Steps:

1. Load Dataset
   - Load the CIC-DDoS2019 dataset (normal and attack traffic).

2. Data Preprocessing
   - Clean data (handle missing values, normalize/standardize features).
   - Split data into training and testing sets (e.g., 70 % train, 30 % test).

3. Feature Selection
   - Apply Fisher Score Algorithm to select the most relevant features.

4. Initialize XGBoost Model
   - Set hyperparameters (learning_rate, max_depth, n_estimators, subsample, gamma, etc.).

5. Train Model
   - Train XGBoost using the training set and selected features.

6. Make Predictions
   - Use the trained model to predict the class (Normal/DDoS) for test data.

7. Evaluate Model
   - Calculate Accuracy, Precision, Recall, F1-Score, AUC, and Confusion Matrix.

8. Output Results
   - Display evaluation metrics and confusion matrix.

9. (Optional) Hyperparameter Tuning
   - Use grid search to optimize hyperparameters.

Schema 3: XGBoost algorithm

XGBoost

XGBoost, short for Extreme Gradient Boosting, is an enhanced version of the Gradient Boosting Machines (GBM) algorithm specifically developed for supervised learning applications. This framework is designed to be scalable and high-performance, with a specific focus on optimizing tree boosting for machine learning. XGBoost improves upon the conventional gradient boosting method by integrating parallel tree construction, resulting in faster model training and enhanced accuracy. The system has features such as optimized cache access patterns, data compression, and data sharing, which collectively enhance the efficiency of the tree boosting process. XGBoost, being an ensemble learning technique, is highly proficient in handling both classification problems, such as credit scoring, and regression tasks. XGBoost differs from traditional gradient boosting by constructing decision trees simultaneously, utilizing its sophisticated features to address intricate real-world issues with greater efficiency. The optimization function to minimize is;

$$L^{(t)} = \sum_{k=1}^{n} l\left(y_k, y_k^{-(t-1)} + \varphi_t(\mathrm{x}_k) + \Omega(\varphi_t)\right) \qquad 2$$

where $l(.)$ is a loss function and $\Omega(\varphi_t)$ is a regularization term that penalizes the complexity of the model. The goal of XGBoost is to find theφ_t, which minimizes the objective function $L^{(t)}$. Table 2 shows the definition of all the parameters used in equation 2. The XGBoost algorithm is shown on algorithm 4 (Schema 4).

Table 2: Definition of parameters

Parameter	Definition
$L^{(t)}$	The objective function (total loss) at the t-th boosting iteration, including regularization penalty.
N	Number of training samples (total data points) in the dataset.
$\sum_{k=1}^{n}()$	Summation across all training samples k = 1,2,...,n.
$l(.)$	Loss function quantifying difference between true label and prediction (e.g., MSE, logistic loss).
y_k	True target value for the k-th training instance.
$y_k^{-(t-1)}$	Prediction for the k-th instance from previous (t-1) iterations (cumulative prediction).
$\varphi_t(\mathbf{x}_k)$	New function (weak learner/tree) added at iteration t, evaluated on x_k; incremental improvement.
$(\mathbf{x}_k)$	Feature vector (input) of the k-th training instance.
$\Omega(\varphi_t)$	Regularization term penalizing complexity of tree phi_t to prevent overfitting (e.g., #leaves, L2).
T	Current boosting round/iteration number (t = 1,2,...,T).

Performance Evaluation

The performance of the SVM model may be assessed by considering metrics such as accuracy, precision, recall, and f1-score. The criterion for each of the measures is determined based on four basic criteria: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN).

i) **True Positive (TP):** Indicates whether the model correctly identified an intrusion when there was indeed an incursion.

ii) **False Negative (FN):** Occurs when the model incorrectly labels an instance record as not a DDOS intrusion, despite it actually being an intrusion.

iii) **False Positive (FP):** Indicates instances where the model incorrectly identifies an intrusion when there is actually no intrusion present.

iv) **True Negative (TN):** Indicates when the model correctly identifies that there is no intrusion and there is indeed no intrusion.

Precision: is a quantitative measure that specifically evaluates the accuracy of identifying positive instances, by determining the proportion of anticipated positives DDOS attack that are actually true. The question it addresses is: "What is the proportion of true positive predictions among all the predicted positive instances?" Equation 3 represents the mathematical formula for the precision metrics.

$$precision = \frac{TP}{TP + FP} \qquad 3$$

Recall: Recall, sometimes referred to as sensitivity or true positive rate, measures the accuracy of predicting actual DDOS intrusion records. It provides the answer to the question: "Out of all the true positive cases, how many were accurately identified as positive?" The formula is displayed in equation 4:

$$Recall = \frac{TP}{TP + FN} \qquad 4$$

F-measure (or F-score): The F1 score is calculated as the harmonic mean of precision and recall. The evaluation of a classification model is conducted in a balanced manner, considering both false positives and false negatives. It is especially advantageous when working with datasets that have an unequal distribution of data. Equation 5 presents the mathematical expression for the F-score.

$$F - Measure = 2 \times \frac{Precision \times Recall}{Precision + Recall} \qquad 5$$

Accuracy: Accuracy is a widely used measure to assess the overall effectiveness of a classification model. Equation 6 demonstrates how the accuracy is calculated by determining the ratio of correctly classified cases (including true positives and true negatives) to the total number of instances:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \qquad 6$$

RESULT AND DISCUSSION

This section outlines the results of the experiment employing Support Vector Machine (SVM) and XGBoost after selecting the relevant features as a feature selection mechanism using Fisher Score algorithm. The models' efficacy is assessed by metrics like accuracy, recall, and F1-score.

Data Scaling

The fundamental purpose of data scaling in this study is to prevent feature dominance, as larger scales might overshadow the learning process and result in biased outcomes. As a result, equal weight is given to all characteristics in the learning process when the dataset is scaled while ensuring faster convergence and stable training for the utilized SVM and XGBoost algorithms. Moreover, the presence of varying units or data types among certain dataset aspects necessitates scaling to maintain a uniform scale. Therefore, to normalize the numerical data to a unit scale, the Standard Scaler from the Scikit-learn module was employed. Additionally, the Label Encoder library was employed to convert the categorical characteristics.

Selected Features

The implementation of FSA was executed using Python within the Anaconda environment, leveraging libraries such as scikit-learn and pandas for data manipulation and analysis. The algorithm processed the preprocessed dataset and ranked the features according to their Fisher Scores. Figure 3 presents a visualization of the features selected.

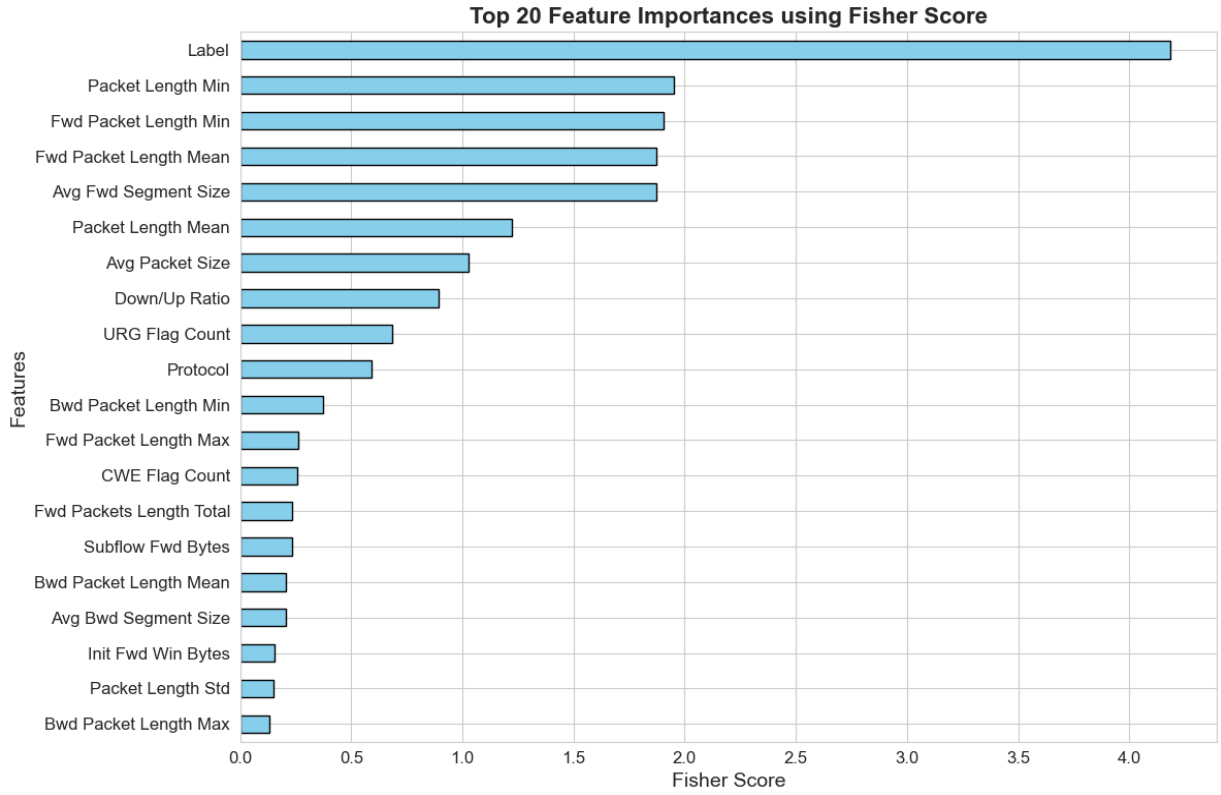


Figure 3: Selected features using Fisher score

Table 3 lists the top 20 features selected by FSA, along with their corresponding importance scores, which indicate their relevance in distinguishing DDoS attack traffic from benign traffic. The table is reproduced below for reference:

Table 3: Feature selected using Fisher ccore

Feature Name	Importance Score
Label	4.184502
Packet Length Min	1.949349
Fwd Packet Length Min	1.905270
Fwd Packet Length Mean	1.872551
Avg Fwd Segment Size	1.872551
Packet Length Mean	1.223484
Avg Packet Size	1.029239
Down/Up Ratio	0.892608
URG Flag Count	0.683811
Protocol	0.593223
Bwd Packet Length Min	0.372296
Fwd Packet Length Max	0.260645
CWE Flag Count	0.256331
Fwd Packets Length Total	0.235340
Subflow Fwd Bytes	0.235340
Bwd Packet Length Mean	0.204430
Avg Bwd Segment Size	0.204430
Init Fwd Win Bytes	0.156676
Packet Length Std	0.152402
Bwd Packet Length Max	0.131278

Percentage Split Technique

In this research, the dataset was split into training and test halves; the training set was utilized to train the SVM and XGBoost algorithms, while the test set was utilized to assess the model's performance. Seventy percent of the dataset were utilized to train the models, while the remaining thirty percent was used to assess the models' performance.

Result Presentation

As stated in the objectives of the study. The performance of the SVM and XG-Boost on the adapted network dataset was be evaluated based on certain performance indicators, such as the accuracy score, precision, recall, AUC, and the F1-Score for cases with and without feature selection using fisher score. Table 4 presents the result of the SVM and XGBoost based on whether an instance is considered DDOS attack or not (0/1) as labelled in the headers of the tables. The metrics result was evaluated using percentages as the degree of measurement.

Table 4 and Figure 4 compare the performance of XGBoost and SVM after applying Fisher Score feature selection for DDoS detection. XGBoost clearly outperforms SVM, achieving 97 % accuracy and an almost perfect AUC of 0.99, while SVM lags with 65 % accuracy and an AUC of 0.65. XGBoost also shows balanced precision and recall for both normal and attack traffic, resulting in an F1-score of 0.98, whereas SVM struggles with poor recall and imbalance between classes. Overall, XGBoost proves to be the more reliable and effective model for real-time DDoS attack detection.

Table 4: Result presentation with feature selection using Fisher score

Model	Accuracy	Precision (0/1)	Recall (0/1)	F1-score (0/1)	AUC
XGBoost	0.97	0.96/0.99	0.99/0.95	0.98/0.98	0.99
SVM	0.65	0.59/1.00	1.0/0.32	0.74/0.47	0.65

SVM Classification Report:

	precision	recall	f1-score	support
0	0.5933	1.0000	0.7447	5865
1	1.0000	0.3156	0.4798	5875
accuracy			0.6575	11740
macro avg	0.7966	0.6578	0.6122	11740
weighted avg	0.7968	0.6575	0.6121	11740

SVM AUC Score: 0.6577872340425532

XGBOOST Classification Report:

	precision	recall	f1-score	support
0	0.9611	0.9986	0.9795	5865
1	0.9986	0.9597	0.9787	5875
accuracy			0.9791	11740
macro avg	0.9798	0.9791	0.9791	11740
weighted avg	0.9799	0.9791	0.9791	11740

XGBOOST AUC Score: 0.9791477752988337

Figure 4: SVM and XGBoost code snippet result with feature selection using Fisher score

Table 5: Result presentation without feature selection

Model	Accuracy	Precision (0/1)	Recall (0/1)	F1-score (0/1)	AUC
XGBoost	0.86	0.86/0.0	1.0/0.00	0.92/0.00	0.50
SVM	0.88	0.87/1.00	1.0/0.23	0.93/0.37	0.61

XGBOOST Classification Report:

	precision	recall	f1-score	support
0	0.8568	1.0000	0.9229	5887
1	0.0000	0.0000	0.0000	984
accuracy			0.8568	6871
macro avg	0.4284	0.5000	0.4614	6871
weighted avg	0.7341	0.8568	0.7907	6871

XGBOOST AUC Score: 0.5

SVM Classification Report:

	precision	recall	f1-score	support
0	0.8861	1.0000	0.9396	5887
1	1.0000	0.2307	0.3749	984
accuracy			0.8898	6871
macro avg	0.9430	0.6153	0.6572	6871
weighted avg	0.9024	0.8898	0.8587	6871

SVM AUC Score: 0.6153455284552846

Figure 5: SVM and XGBoost code snippet result with feature selection

Table 5 and Figure 5 show that removing feature selection significantly reduces the performance of both XGBoost and SVM in DDoS detection. Without the Fisher Score, XGBoost achieves 86% accuracy but completely fails to identify attacks, with an AUC of 0.50, equivalent to random guessing. SVM performs slightly better with 88 % accuracy and an AUC of 0.61, yet still misses most attacks due to poor recall. These results highlight that feature selection is essential, as its absence leads to major declines in accuracy, recall, and overall reliability for both models in detecting DDoS attacks.

Confusion matrix

Additional to rigorously access the performance of individual algorithms the confusion matrix was considered. Here, the confusion matrix serves as a valuable tool for understanding the performance of an algorithm in the classification of DDOS attack or benign traffic as shown in Figure 6.

From the confusion matrix in Figure 6, the SVM model correctly identified 5887 normal instances and 227 attacks but misclassified 757 attacks as normal, showing difficulty in detecting class 1. It had no false positives, indicating accurate classification of normal traffic. In contrast, XGBoost correctly detected all normal instances but failed to identify any attacks, misclassifying all 984 as normal. This reveals a strong bias toward class 0 and an inability to detect attacks, resulting in zero recall for class 1 without feature selection.

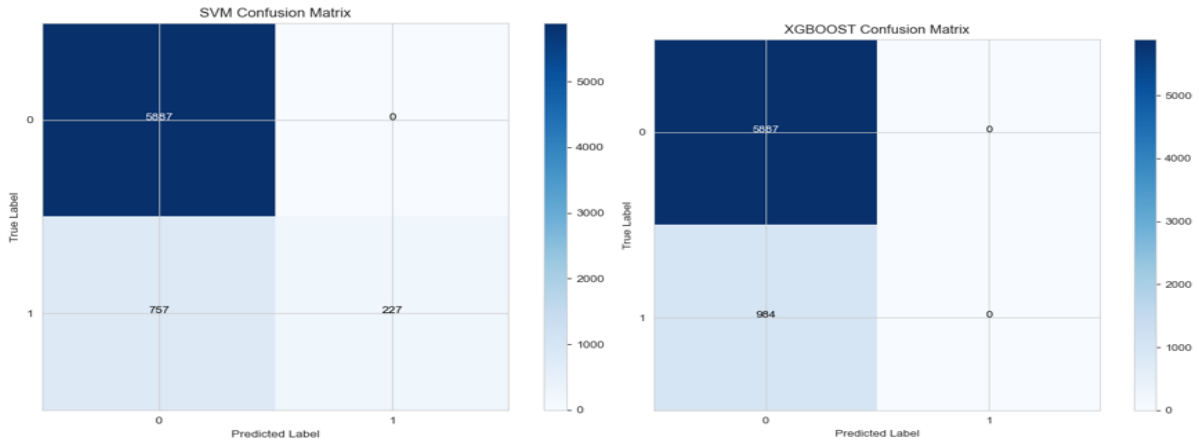


Figure 6: SVM and XGBoost confusion result without feature selection

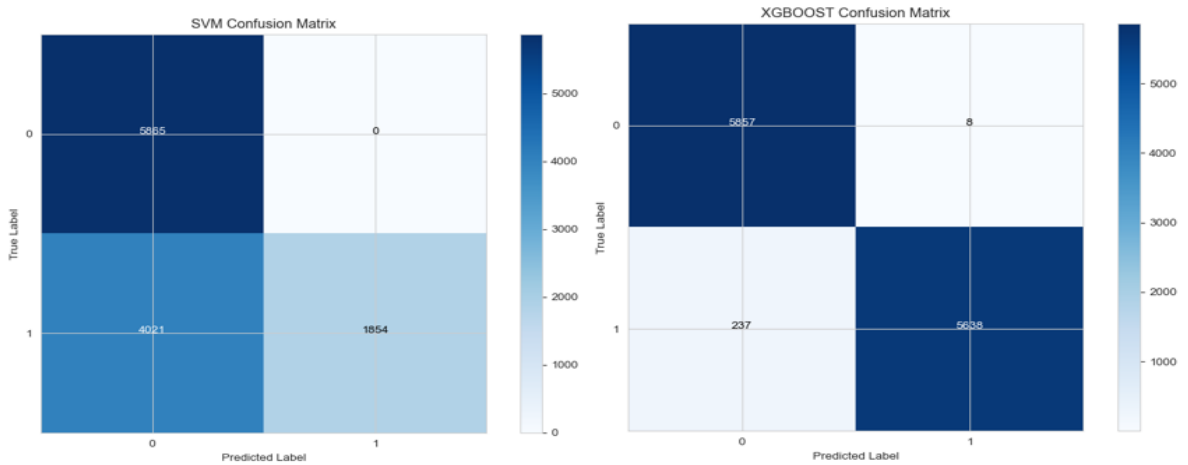


Figure 7: SVM and XGBoost confusion result without feature selection

Applying Fisher Score feature selection greatly improved the performance of both SVM and XGBoost, as shown in Figure 7. SVM correctly classified more positive cases than before but still produced many false negatives, indicating continued difficulty in detecting attacks. In contrast, XGBoost showed a major improvement, accurately identifying most positive and negative instances with only a few errors. This demonstrates that feature selection significantly enhances XGBoost's recall and overall detection ability, while SVM benefits moderately but still needs further refinement.

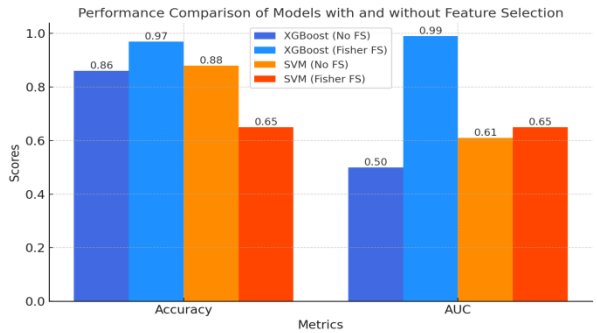


Figure 8: Result comparison on bar chart

Graphical Comparison

The bar chart from Figure 8 visually compares the performance of XG-Boost and SVM models before and after applying feature selection using the Fisher Score method. The key metrics displayed include accuracy and the area under the curve (AUC), which are essential for evaluating classification performance.

Results from Tables 4 and 5, along with confusion matrices from Figures 6 and 7, strongly prove that there is a significant difference between the performances of the XGBoost and SVM models in DDoS detection, with and without feature selection using the Fisher Score. XGBoost performed much better than SVM in almost all the evaluation metrics, and this gap was wider after applying feature selection. The following reasons may be responsible for such a huge observed performance difference: non-linearity handling, interaction of features, importance of feature selection, and adaptability to the underlying data distribution.

1. Non-Linearity: The key power of XGBoost in modeling complex and non-linear data relationships can be derived through decision trees. In an additive manner, the gradient boosting algorithm used by XGBoost builds trees; it means every new tree corrects errors developed in the past by a previous tree. It thus enables XGBoost to learn effectively from residuals,

hence capturing the intricate patterns in data, more so when the relationships between features are nonlinear.When dealing with a non-linear dataset, SVM could address this issue through kernel functions, for example, Radial Basis Function (RBF). Nevertheless, this model requires hyperparameters such as kernels and gamma values, which demand more tuning. Therefore, based on this experiment, where SVM failed in terms of recall and AUC metrics, this suggests that this model might not be well tuned, especially in handling kernels, which might sometimes influence more in distinguishing, hence causing this model to fail more, especially in high-dimensional spaces as is often experienced in a DDoS network attack scenario.

2. Feature Interactions and Importance: One of the strong advantages that make XGBoost different from other models is its ability to effectively address feature interaction. This is because XGBoost, through the ensemble technique of decision trees, is capable of recognizing interaction within various features. In this regard, to improve model performance to detect DDoS attacks, feature selection through the Fisher score technique is highly reliable. This is justified through a remarkable increment and rise in accuracy and AUC in Table 4 when applying feature selection to the model, whereby recall and precision values are highly balanced for XGBoost, with an F1-score of 0.98. On the other hand, the SVM model is not that good at modeling complex interactions between features unless their space is explicitly mapped onto a higher or different space via kernel functions. This generational model improves SVM's recall and F1-score with the Fisher Score feature selection system, shown in Table 4, but it still suffers from class imbalance and an inability to learn key interaction patterns; this can be seen from the significantly lower AUC and the large gap in recall between classes 0 and 1, Normal versus DDoS. The very large difference in recall values between Normal - with 1.00 - and DDoS - with 0.32 - indicates that even with feature selection, SVM is incapable of distinguishing between benign traffic and attacks.

3. Effect of Feature Selection: The selection of features by Fisher Score enhances the ability of XGBoost to detect normal and attack traffic correctly, which is clearly depicted in Table 4. This model benefits from reduced dimensionality and becomes highly effective with much better precision, recall, F1-score, and an AUC of 0.99. As seen in the figure above, without the usage of any feature selection method, the accuracy of XGBoost reduces drastically, as evident from the values in Table 5, where the AUC value reduces to 0.5. Although the feature selection process is improving the values of the recall metric of SVM and reducing the false positives, as demonstrated in Table 4, the issue of class imbalance is not resolved. As already mentioned, this model is very sensitive to class imbalances, meaning that the repeated failure of the model to perform a proper identification of attacks, as demonstrated in Table 5, is not surprising. Since this model has a very low recall value in Table 5, it is evident that the limitations of this model regarding the identification of classes related to DDoS attacks persist. Although this model has a better accuracy rating of about 88 %, as demonstrated before, this is not without a price, meaning that this model is significantly impacted by the class imbalance issue.

4. Class Imbalance and Model Adaptability: Another reason why XGBoost performs better is its robust handling of class imbalance. XGBoost is capable of dealing with data and improving its performance through boosting with its ability to concentrate on certain data points it finds troublesome to classify. This becomes helpful when dealing with class imbalance as is seen with DDoS attacks. The classes are imbalanced since DDoS attacks are relatively rare. XGBoost can thus be improved using weight values or even adjusting parameters like 'scale_pos_weight' to perform well even when both classes are imbalanced, as seen with the precision and recall values of both classes in Table 4. The problem with SVM is that it is affected when there is class imbalance because it aims at maximizing the class boundaries. This might make SVM not work effectively when there is class imbalance. This is demonstrated when we apply class imbalance to SVM and find that there is difficulty with attacks being classified as normal. Even after applying feature selection to SVM, we find that there is difficulty with detecting the class for DDoS attacks, as demonstrated from the fact that SVM manages to classify only 0.32 of class DDoS.

5. Model Complexity and Tuning: The XGBoost algorithm is complex in its concept of using ensemble learning in a way that combines many weak learners–trees–in its approach to training a robust learner. The hyperparameters of this model, such as learning rates and maximum depth of trees and estimators, are some of the features of this approach and can be adjusted to optimize this algorithm's performance. The nature of this algorithm allows it to handle more complex data trends with additional trees in its approach to boosting this learner trained on data. Although SVM is simpler in architecture with only a single hyperplane between two classes of data points, its performance on complex data sets requires proper tuning of its kernel functions for proper performance. Although in this research SVM's tuning is not clear to obtain proper results as far as DDoS intrusion detection is concerned.

Model Deployment

The deployment of the machine learning model for DDoS attack detection marks a critical phase in transitioning from research to practical application. This section details the development process of the web-based application, the technologies used, the structure of the app, and a step-by-step guide on how users can interact with it to detect DDoS attacks in real-time. The deployed application leverages the trained XGBoost model, which demonstrated superior performance with an accuracy of 97 % and an AUC of 0.99 after feature selection using the FSA. Figure 9 below show the application interface.

Figure 9: DDos attack detection application

Development of the Web Application

The web application was developed to provide an accessible and user-friendly interface for network administrators and cybersecurity professionals to detect potential DDoS attacks. The app was built using a combination of Python-based frameworks and front-end technologies to ensure seamless integration of the machine learning model with a graphical user interface (GUI).

Framework: The backend of the application was developed using Flask, a lightweight and flexible Python web framework. Flask was chosen for its simplicity and ability to handle HTTP requests efficiently, making it ideal for integrating the machine learning model with the web interface.

Model Integration: The trained XGBoost model, which was saved as a serialized file (e.g., using Python's joblib or pickle library) after training in the Anaconda environment, was loaded into the Flask application. This allowed the model to make predictions on new input data provided by users.

Data Preprocessing: A preprocessing pipeline was implemented within the Flask backend to handle incoming user inputs. This pipeline mirrors the preprocessing steps applied during model training, including data scaling (using StandardScaler from scikit-learn) and label encoding (using LabelEncoder). The selected features identified by the Fisher Score Algorithm were used to ensure that only the most relevant attributes are processed by the model.

Prediction Logic: Upon receiving user inputs, the Flask backend processes the data, passes it through the XGBoost model, and returns a binary prediction (0 for benign traffic, 1 for DDoS attack traffic). The prediction is then rendered on the web interface for the user to view.

Deployment Environment

The application was deployed locally on a Windows operating system, consistent with the development environment. The Flask app was hosted on a local server (e.g., http://127.0.0.1:5000, as indicated in the screenshot (see Appendix)), making it accessible via a web browser.

CONCLUSION

This paper proposes the development and evaluation of the Support Vector Machine and Extreme Gradient Boosting models for the detection of Distributed Denial of Service attacks using the CIC-DDoS2019 dataset. The methodology involves Fisher Score feature selection and a systematic train-test split for model assessment. Experimental results have shown that feature selection significantly improved XGBoost performance, reaching an accuracy of 97 % and an AUC of 0.99, while for SVM, its performance decreased to an accuracy of 65 %, although the AUC was marginally improved. It is clear from the results that XGBoost outperforms SVM in DDoS detection, especially when feature selection using Fisher Score is applied. XGBoost can handle non-linearity, capture feature interactions, and deal with class imbalance, making it the more suitable model for real-time DDoS detection. While SVM performs reasonably well in terms of accuracy, its inability to properly detect the minority class (DDoS attacks) due to poor recall and imbalance between classes limits its effectiveness. Feature selection plays a great role in improving the performance of both models, whereby XGBoost has benefited the most to reinstate the importance of feature selection in improving machine learning model performance in security applications.

The developed DDoS detection framework demonstrates strong applicability across several cybersecurity domains, including enterprise network protection, critical infrastructure defense, and cloud security enhancement. Its deployment can facilitate proactive intrusion detection, safeguard critical assets, and preserve data integrity in dynamic computing environments.

It is recommended that future research focus on advanced feature engineering, ensemble-based modeling, and adaptive learning mechanisms to further improve the robustness, scalability, and responsiveness in real time. Exploring hybrid or reinforcement learning approaches could further enable the system to dynamically adapt to evolving network threats and improve overall detection efficiency in real-world cybersecurity applications.

REFERENCES

Ajagbe, S. A., Awotunde, J. B. and Florez, H. (2024). Intrusion Detection: A Comparison Study of Machine Learning Models Using Unbalanced Dataset. *SN Computer Science*, 5(8), 1028. https://doi.org/10.1007/s42979-024-03369-0

Alhasawi, Y. and Alghamdi, S. (2024). Federated Learning for Decentralized DDoS Attack Detection in IoT Networks. *IEEE Access*, 12, 42357–42368. https://doi.org/10.1109/ACCESS.2024.3378727

AlSaleh, I., Al-Samawi, A. and Nissirat, L. (2024). Novel machine learning approach for DDoS cloud detection: Bayesian-based CNN and data fusion enhancements. *Sensors*, 24(5), 1418. https://doi.org/10.3390/s24051418

Altulaihan, E., Almaiah, M. A. and Aljughaiman, A. (2024). Anomaly detection IDS for detecting DoS attacks in IoT networks based on machine learning algorithms. *Sensors*, 24(2), 713. https://doi.org/10.3390/s24020713

Asaju, La'aro Bolaji, Peter Bamidele Shola, Nwadike Franklin, and Hambali Moshood Abiola. (2017). Intrusion detection system on a computer network using an ensemble of randomizable filtered classifier. *K-Nearest Neighbor Algorithm*, 2, 550–553.

Baghdad, Iraq, Abdulrahman, N. F. and Jit Singh, M. S. (2025). Deep learning approaches for DDoS attack detection in communication networks and IoT: A comprehensive review. *Journal Kejuruteraan*, 37(1), 323–333. https://doi.org/10.17576/jkukm-2025-37(1)-22

Berbiche, N. and El Alami, J. (2024). For robust DDoS attack detection by IDS: Smart feature selection and data imbalance management strategies. *Ingénierie Des Systèmes d Information*, 29(4). https://doi.org/10.18280/isi.290401

Berhanu, B. and Assamnew, F. (2024). *Application Layer Ddos Attack Classification Using Deep Learning*. https://doi.org/10.2139/ssrn.4687852

Brown, E., Fisher, J., Hudon, A., Colston, E. and Lu, W. (2024). Multiclassification analysis of volumetric, protocol, and application layer DDoS attacks. In L. Barolli (Ed.), *Advanced Information Networking and Applications* (Vol. 204, pp. 401–413). Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-57942-4_39

Hadeel S. Obaid and Esamaddin H. Abeed (2020). *DoS and DDoS Attacks at OSI Layers*. https://doi.org/10.5281/ZENODO.3610833

Hambali, Moshood Abiola and Oforjetu Chukwudi Peter (2024). *An Android Malware Detection System Based on Hybrid Artificial Neural Network and Decision Tree Algori.*, 8(2), 45-68.

Jawahar, A, J., Kaythry, P, Vinoth Kumar, C, Vinu, R., Amrish, Bavapriyan, K. and Gopinaath, V. (2024). DDoS mitigation using blockchain and machine learning techniques. *Multimedia Tools and Applications*, 83(21), 60265–60278. https://doi.org/10.1007/s11042-023-18028-4

Khan, S. A., Hussain, S. I. and Iqbal, J. (2025). From Signatures to AI: A Comprehensive Review of DDoS Detection Strategies in IoT & SDN. *Int. J. on Robotics, Automation and Sciences*, 7(1), 19–26. https://doi.org/10.33093/ijoras.2025.7.1.3

Li, J., Luo, T., Zhang, B., Chen, M. and Zhou, J. (2024). IMOABC: An efficient multi-objective filter–wrapper hybrid approach for high-dimensional feature selection. *J. of King Saud University - Computer and Information Sci.*, 36(9), 102205. https://doi.org/10.1016/j.jksuci.2024.102205

Mahboubi, A., Luong, K., Aboutorab, H., Bui, H. T., Camtepe, S., Ansari, K. and Barry, B. (2025). The evolving threat landscape of botnets: Comprehensive analysis of detection techniques in the age of artificial intelligence. *Internet of Things*, 33, 101728. https://doi.org/10.1016/j.iot.2025.101728

Merkebaiuly, M. (2024). Overview of distributed denial of service (DDoS) attack types and mitigation methods. *Inter. Conf.*, 43(193), 494–508. https://doi.org/10.51582/interconf.19-20.03.2024.048

Pakmehr, A., Aßmuth, A., Taheri, N. and Ghaffari, A. (2024). DDoS attack detection techniques in IoT networks: A survey. *Cluster Computing*, 27(10), 14637–14668. https://doi.org/10.1007/s10586-024-04662-6

Pang, G., Shen, C., Jin, H. and Van Den Hengel, A. (2023). Deep weakly-supervised anomaly detection. *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1795–1807. https://doi.org/10.1145/3580305.3599302

Patel, M., Amritha, P. P., Sudheer, V. B., & Sethumadhavan, M. (2024). DDoS attack detection model using machine learning algorithm in next generation firewall. *Procedia Computer Science*, 233, 175–183. https://doi.org/10.1016/j.procs.2024.03.207

Ramakrishnan, K., Lutui, R. and Vaipulu, T. (2025). Enhancing cyberattack detection using machine learning techniques in intrusion detection systems. *2025 IEEE 35th International Telecommunication Networks and Applications Conference* (ITNAC), pp. 1–8. https://doi.org/10.1109/ITNAC66378.2025.11302561

Raza, M. S., Sheikh, M. N. A., Hwang, I.-S. and Ab-Rahman, M. S. (2024). Feature-selection-based DDoS attack detection using AI algorithms. *Telecom*, 5(2), 333–346. https://doi.org/10.3390/telecom5020017

Sambangi, S. and Gondi, L. (2020). A machine learning approach for DDoS (distributed denial of service) attack detection using multiple linear regression. *The 14th International Conference on Interdisciplinarity in Engineering— INTER-ENG*, 51. https://doi.org/10.3390/proceedings2020063051

Santos-Neto, M. J., Bordim, J. L., Alchieri, E. A. P. and Ishikawa, E. (2024). DDoS attack detection in SDN: Enhancing entropy-based detection with machine learning. *Concurrency and Computation: Practice and Experience*, 36(11), e8021. https://doi.org/10.1002/cpe.8021

Setia, H., Chhabra, A., Singh, S. K., Kumar, S., Sharma, S., Arya, V., Gupta, B. B. and Wu, J. (2024). Securing the road ahead: Machine learning-driven DDoS attack detection in VANET cloud environments. *Cyber Sec. and Applic*, 2, 100037. https://doi.org/10.1016/j.csa.2024.100037

Setitra, M. A., Fan, M., Benkhaddra, I. and Bensalem, Z. E. A. (2024). DoS/DDoS attacks in software defined networks: Current situation, challenges and future directions. *Comp. Comm.*, 222, 77–96. https://doi.org/10.1016/j.comcom.2024.04.035

Songa, A. V. and Karri, G. R. (2024). An integrated SDN framework for early detection of DDoS attacks in cloud computing. *Journal of Cloud Computing*, 13(1), 64. https://doi.org/10.1186/s13677-024-00625-9

Uddin, R., Kumar, S. A. P. and Chamola, V. (2024). Denial of service attacks in edge computing layers: Taxonomy, vulnerabilities, threats and solutions. *Ad Hoc Networks*, 152, 103322. https://doi.org/10.1016/j.adhoc.2023.103322

Xie, B., Wang, Y., Wen, G. and Xu, X. (2023). Application-layer DDoS attack detection using explicit duration recurrent network-based application-layer protocol communication models. *Int. J. of Intelligent Systems*, 1, 2632678. https://doi.org/10.1155/2023/2632678

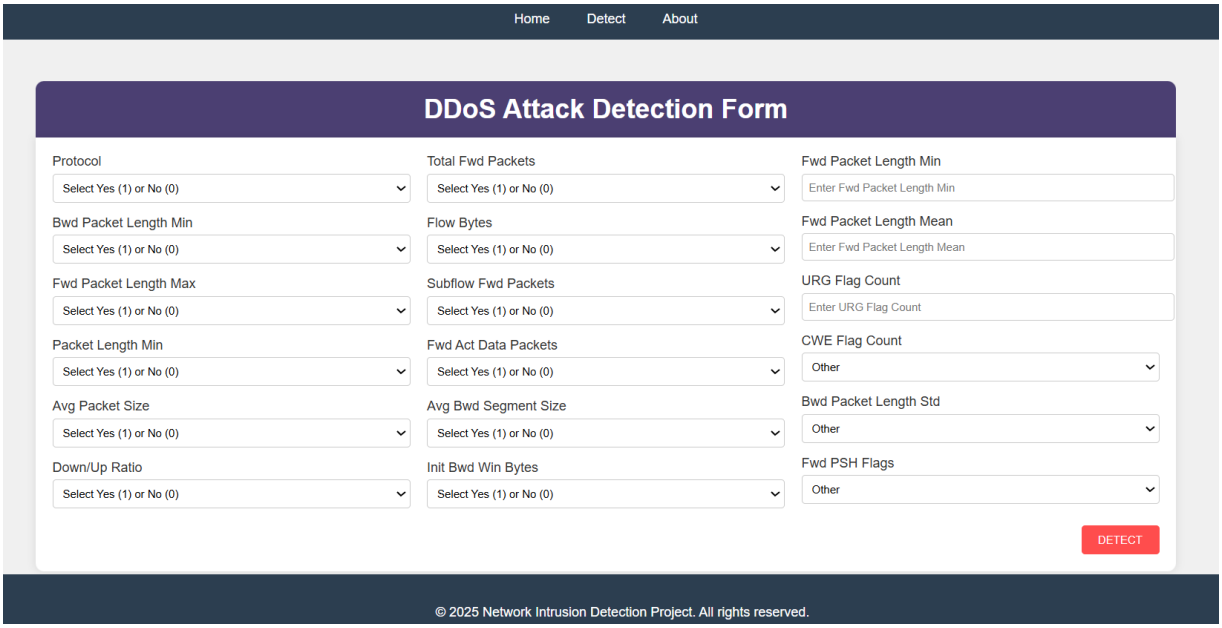
Xu, K., Li, Z., Liang, N., Kong, F., Lei, S., Wang, S., Paul, A. and Wu, Z. (2024). Research on multi-layer defense against DDoS attacks in intelligent distribution networks. *Electronics*, 13(18), 3583. https://doi.org/10.3390/electronics13183583

APPENDIX

How Users Can Use the Application

Below is a step-by-step guide for users:

Access the Application: Open a web browser and navigate to the URL where the app is hosted. In the development environment, this is http://127.0.0.1:5000, as shown in the screenshot provided below:



DDoS Attack Detection Form

The user will be greeted with the "DDoS Attack Detection Form" interface.

Input Network Traffic Data: The form contains fields corresponding to the 20 features selected by the Fisher Score Algorithm. Users must provide values for each feature based on the network traffic they wish to analyze.

For binary features (e.g., "Protocol," "Total Fwd Packets"), select either "YES (1)" or "NO (0)" from the dropdown menus.

For numerical features (e.g., "Fwd Packet Length Min," "URG Flag Count"), enter the appropriate values in the text fields. These values can be obtained from network monitoring tools or packet analysis software like Wireshark, which can extract such metrics from captured traffic.

Submit the Form: Once all fields are filled, click the blue "Detect" button at the bottom right of the form.

The Flask backend will process the inputs, apply the necessary preprocessing (scaling and encoding), and pass the data through the XGBoost model to generate a prediction.

View the Prediction: After submission, the app will display the prediction result on the screen. The result indicates whether the input traffic is classified as benign (0) or a DDoS attack (1).

A prediction of "0" suggests that the traffic is benign and does not indicate a DDoS attack.