International Journal of Safety and Security Engineering
Vol. 14, No. 6, December, 2024, pp. 1859-1869
Journal homepage: http://iieta.org/journals/ijsse

A Fog Computing-Based Machine Learning Framework for DDoS Attacks Detection: Balancing Offline and Real-Time Analysis for IoT Data

Eman Karkawi Kareem[1], Mehdi Ebady Manaa[1,2*]

[1] Department of Information Networks, College of Information Technology, University of Babylon, Hillah 51002, Iraq
[2] Intelligent Medical System Department, College of Sciences, Al-Mustaqbal University, Hillah 51001, Iraq

Corresponding Author Email: mahdi.ebadi@uomus.edu.iq

https://doi.org/10.18280/ijsse.140620

Received: 26 September 2024
Revised: 9 December 2024
Accepted: 19 December 2024
Available online: 31 December 2024

Keywords:
Internet of Things (IoT), fog computing, DDoS attack, IoT security, lightweight, speck encryption, real-time

ABSTRACT

The rapid proliferation of the Internet of Things (IoT) has significantly increased the risk of Distributed Denial of Service (DDoS) attacks, threatening the reliability, availability, and security of services and infrastructure. To address these challenges, this study introduces a novel, integrated framework combining fog computing, machine learning, and lightweight encryption to enhance offline training and real-time detection of DDoS attacks in IoT environments. Our approach differs from existing methods by leveraging an offline phase for model training on recent DDoS patterns. This enables accurate, scalable detection when the model is deployed in the online fog layer. This two-phase strategy ensures timely and resource-efficient threat mitigation. In the offline phase, we extract four critical packet features (Src_IP, Port_IP, Dst_IP, Dst_Port) from the CIC-DDoS2019 and Edge-IIoTset datasets. We then apply Chi-Square and entropy-based feature analysis, followed by synthetic minority oversampling (SMOTE), to address class imbalance. Three core classifiers—Random Forest (RF), Support Vector Machine (SVM), and Decision Tree (DT)—are trained to detect a variety of DDoS attacks (SYN, UDP, HTTP, and TCP) with high accuracy. The online phase deploys the trained model at the fog layer, employing the Speck lightweight encryption algorithm and Elliptic-Curve Diffie–Hellman (ECDH) for secure end-to-end communication. A voting mechanism among classifiers enhances detection reliability. The experiments proved that the framework achieves almost perfect detection accuracy (100% in most scenarios), surpassing current methods in accuracy, scalability, and applicability in resource-limited IoT environments. In addition, network performance metrics (throughput, latency, execution time, response time) confirm the solution's efficiency. This research provides a secure, adaptive, high-performance DDoS attack detection system for IoT systems, laying the foundation for future studies to expand attack coverage, improve real-time performance, and investigate more robust encryption methods.

1. INTRODUCTION

The Internet of Things (IoT) has rapidly emerged as a transformative technological paradigm, integrating the physical and digital worlds to deliver advanced services across industry, healthcare, transportation, and other critical domains. By enabling pervasive connectivity and intelligent data exchange, IoT systems have introduced unprecedented convenience and automation. However, these benefits come with significant security challenges due to the inherently resource-constrained nature of IoT devices (e.g., limited memory and CPU) and their continuous connectivity to open and diverse networks [1, 2].

One of the core difficulties in IoT ecosystems is efficiently handling the massive volume of generated data while ensuring low latency and robust security. Traditional cloud-centric architectures often encounter high delays, bandwidth bottlenecks, and scalability issues as data travels between endpoints and remote servers. To overcome these limitations, fog computing has been introduced to process and store data closer to the network edge, thus reducing latency and enhancing Quality of Service (QoS) [3, 4]. Despite these advantages, fog computing inherits various security and privacy vulnerabilities from the cloud, including various attacks such as message replay, man-in-the-middle attacks, identity spoofing, and more [5]. These threats are especially concerning given IoT networks' increasing complexity and connectivity.

Among the numerous cyber threats facing IoT systems, DDoS attacks are particularly severe. They flood targeted networks or services with a massive volume of traffic, exhausting resources and causing disruptions that can undermine service availability [6, 7]. With IoT devices often lacking powerful computational capabilities, distinguishing benign from malicious traffic in real-time becomes a formidable challenge. Although Machine Learning (ML) algorithms have shown promise in detecting known DDoS attack patterns, conventional methods often struggle to

identify novel or less common attacks, especially under conditions of data imbalance and limited device resources [8].

Security measures that ensure data privacy and integrity are equally critical in IoT environments. Sensitive information transmitted among IoT devices must be protected against unauthorized access [9]. Lightweight encryption and authentication schemes are essential for meeting IoT networks' real-time constraints and low-power requirements. Elliptic Curve Cryptography (ECC), for instance, has gained prominence for secure key sharing and digital identities [10]. The Diffie-Hellman (DH) algorithm, especially when combined with elliptic curves (ECDH), facilitates secure key exchange without transmitting the secret key directly over the network [11, 12]. Such cryptographic methods are vital for establishing trust and confidentiality in IoT ecosystems.

In parallel, the demand for lightweight encryption algorithms suitable for resource-constrained environments has led to the adoption of ciphers like Speck. Speck is a family of lightweight, symmetric block ciphers that enable efficient and secure encryption on IoT devices [13]. Unlike traditional substitution-permutation networks, Speck relies on operations such as XOR, rotations, and modular additions to achieve cryptographic strength with minimal resource overhead. Integrating lightweight encryption ensures that data confidentiality does not compromise latency or power consumption in fog-based IoT scenarios.

Building upon these insights, this work aims to enhance real-time DDoS detection and mitigation in a fog computing environment for IoT networks. Our proposed framework operates in two phases. The offline phase involves training robust detection models using recent IoT datasets (CIC-DDoS2019 and Edge-IIoTset) to handle various DDoS attack patterns. We apply Chi-Square and entropy-based feature selection to improve classifier performance and employ SMOTE to address class imbalance. Machine learning algorithms such as SVM, DT, and RF are utilized and evaluated using accuracy, precision, recall, and F1-score metrics. The trained model is integrated into the fog layer in the online phase, where ECDH ensures secure end-to-end authentication. At the same time, the Speck algorithm provides lightweight encryption for data in transit. This dual-layered approach enables efficient, reliable, and secure DDoS detection in real-time, with network performance indicators such as latency, response time, and throughput also assessed to confirm system scalability and robustness.

In short, our research addresses the urgent need for a secure, accurate, and low-latency solution for detecting denial-of-service attacks in IoT environments. By integrating machine learning-based anomaly detection, fuzzy computing engineering, lightweight cryptographic primitives, and powerful feature engineering, we are developing the current state of IoT Cybersecurity and providing a foundation for future improvements in dealing with emerging threats.

2. RELATED WORK

This section will shed light on various studies discussed here. Many researchers use symmetric algorithms to detect DDoS attacks, using an algorithm that combines data mining and machine learning. A data encryption algorithm is proposed speck algorithm.

Al-Razaq et al. [14] proposed an advanced system for classifying spam emails based on machine learning and deep learning. The study focused on using a hybrid model combining deep neural networks (DNN) and neural bypass networks (CNN), where the model was applied to a dataset of 5,172 e-mails, with the most common 3,000 words used as the main features. To determine the top 500 features, I used the random Forest algorithm. The proposed model showed an overall accuracy of 99.8% in classifying messages, with a high detection rate of spam reaching 99.81% and a very low false positive rate. The proposed model also outperformed other models, such as random forests (95.45%), NB (96.47%), and traditional CNN networks (96.39%). This development reflects the importance of combining deep learning technologies to improve the performance of spam detection systems and reduce the negative effects of spam.

Mallampati and Hegde [15] proposed the adoption of a hybrid model combining deep bypass neural networks (CNN) and a cost-sensing supporting machine algorithm (SVM), to improve the classification of spam, whether text or image. The model addresses the problem of unbalanced data distribution, emphasizing minimizing the cost of errors. The methodology was tested on two data sets, Spambase (4601 emails) and Wish (1730 images), where the model achieved an accuracy of 98.05% and an AUC value of 99.01%, surpassing traditional models such as AdaBoost and random Forest. These results confirm the effectiveness of the hybrid model in recognizing spam with low error rates, with a recommendation to increase data diversity in future studies to ensure better generalization.

Lawal et al. [16] introduced a framework for mitigating DDoS attacks within an IoT computing environment using fog computing to achieve swift and precise detection. This framework leverages an anomaly-based mitigation strategy that employs a k-NN classification algorithm with a specialized database. The database maintains signatures of previously identified attacks, facilitating quicker detection during recurring attacks. The proposed k-NN classifier was evaluated using the CIC-DDoS2019 dataset. The experimental outcomes revealed that the k-NN classifier successfully identified DDoS attacks with high accuracy, surpassing other ensemble classifiers in binary classification tasks. Specifically, the k-NN classifier attained an accuracy rate of 99.99%, while the DT and NB classifiers achieved 99.88% and 95.55%, respectively. Future research aims to implement the framework on existing fog computing platforms to validate the proposed methodology further.

Machine learning techniques were adopted to contribute to developing an effective model that detects and classifies one of the most important DDoS attacks based on the analysis of the extent of valid data traffic in the network. Singh Samom and Taggu [17] pointed to the latest dataset, CIC-DDoS2019, which includes modern reflective DDoS attacks. The proposed system is evaluated based on four performance evaluation parameters: accuracy, predictive accuracy, recall, and F1 value, as well as prediction time. Experimental results showed that the model based on the RF classifier achieved an outstanding performance of 99.927%, superior to other classification algorithms. In the future, a framework based on this method of detecting and processing traffic attacks in real-time is expected to be developed to address security challenges more effectively.

The performance values of three techniques of group learning, namely packing, reinforcement, and stacking, as well as three traditional techniques in machine learning, namely the nearest neighbor algorithm K, the RF, and NB, to detect intrusion in smart grid networks, as proposed by Khoei et al.

[18]. Use the CIC-DDoS2019 standard as the evaluation criterion. The stack-based group learning technique showed the best performance results of exploit attacks compared to all other classifiers, achieving TPR rates of 96%, FPR of 1%, FNR of 0.7%, and accuracy of 97.3%. Current systems face many limitations, such as low detection rates and high false alarm rates. Based on this, many studies have focused on addressing these issues.

Yungaicela-Naula et al. [19] proposed the implementation of a modular and flexible Software-Defined Networking (SDN)-based architecture to detect DDoS attacks at both the transport and application layers. It used multiple frameworks for ML and DL. By exploring various ML/DL approaches, the study identified the best models under different types of attacks and conditions. Among the machine learning technologies, the KNN and SVM models successfully detected high-volume attacks using the CIC-DDoS2019 dataset with an accuracy exceeding 99.77%. The RF achieved an accuracy of 96.36%. Future work involves integrating a scalability component and a mitigation module into the proposed architecture. Through the integration of various algorithm performances, DDoS attacks have been classified into multiple categories using machine learning algorithms proposed by Mishra in 2022. Each type of attack was detected and validated using specific characteristic criteria. To identify multi-category cyber threats associated with DDoS attacks, a comprehensive analysis of diverse machine learning algorithms was conducted. The RF and SVM classifiers demonstrated the highest accuracy, each achieving an accuracy rate of 99.99%. In contrast, the NB classifier attained an accuracy of 99.98%, while the DT classifier recorded an accuracy of 99.89% using the CIC-DoS2019 dataset. Future research indicates that this approach could be extended to target, classify, and predict various other DDoS attacks [20].

Ogini et al. [21] found that a design-based model is proposed to detect and prevent DDoS attacks by controlling malicious traffic and reducing it on the network within the computing of IoTs, based on machine learning techniques. Their experiments utilized five ensemble classifiers tested on the latest DDoS attack dataset, CIC-DDoS2019. The results showed that the DT algorithm for the combined encapsulation classifier contributed to improving the classification accuracy of data traffic by 99.75%, which could further enhance the model's performance and enable real-time deployment in IoT environments. This study specifically targets identifying both 'IoT attacks' and 'DDoS attacks' using the CIC-DDoS2019 dataset provided by the Canadian Institute of Cybersecurity. Garg suggested this [22]. A boosting and non-boosting approach was used to identify the attacks. The boosting approach was found to be suitable for identifying attacks. LGBM is the most efficient of the two boosting methods, with an accuracy of 94.79%.

Tareq et al. [23] suggested a model for detecting cyber-attacks through a multiclass classification approach—the evaluation commenced by assessing the performance of the Edge-IIoTset dataset. Comparative analysis was conducted to discern various cyber-attacks. The highest accuracy obtained was 94.94%, which achieved training and validation accuracy. After 34 epochs, the spill can be seen, where the introduction Time algorithm was used in the Edge-IIoTsetset dataset because one of the challenges faced by using memory in the experiments was this dataset. Hence, the classes used weights instead of SMOTE as the experiments were within the limited possibilities of the CPU with celibacy regarding the use of memory and test time.

AI-based learning models enable security analysts to understand better the nature of cyber threats and devices and more effective mitigation strategies. This study focused on preprocessing, analyzing, and evaluating data collected from digital sensors within an IoT system to identify potential vulnerabilities associated with IoT and IIoT network protocols has been proposed by Hamza et al. [24]. To this end, various machine learning algorithms were evaluated, including KNN, achieving 90.3% accuracy; DTC, 92.5% accuracy; (LR, 81.5% accuracy; SVM, 84.3% accuracy; and RF, 94.1% accuracy, using the publicly available Edge-IIoTset malware detection dataset. The experimental results clearly show that RF outperformed other algorithms, achieving an impressive 94% accuracy in malware detection. Future research would be on investigating the efficacy of transfer learning techniques in the context of malware detection.

Laiq et al. [25] aimed this study to identify normal or malicious DDoS attacks in the IoT terminal network (DDOS traffic). The proposed study used XGBoost, a combination of SVM, DT, and NB, through strict voting to predict normal and harmful traffic using the Edge-IIoTset dataset. In addition, the results indicate that the strict voting classifier achieved 88.7% accuracy of XGBoost and 99.88% and outperformed the strict voting group classifier by 11%. Future research will be on investigating the effectiveness of transformational learning techniques in the context of malware detection. Present a Federated Learning (FL) method to detect intrusions and defend IoT networks. To test the method's efficacy, we ran thorough experiments on Edge-IIoTset. The suggested intrusion detection model's accuracy (92.49%) is close to the standard centralized ML models' (93.92%) utilizing the FL approach, proving its dependability and effectiveness. Future directions are based on making the system more reliable for the case study where nodes at the edges are harmful to the network [26].

3. THE PROPOSED SYSTEM

In the proposed framework, the system operates through an integrated two-phase approach: an offline phase for preprocessing, feature extraction, and model training and an online phase for real-time detection and response. This design ensures that robust models are trained on historical data and refined feature sets (using Chi-Square, Entropy, and SMOTE) before being deployed at the fog layer. By separating these two phases, the system combines the analytical depth of offline processing with the immediacy and security needs of online DDoS detection.

The research technique comprises two distinct layers: the first is dedicated to IoT devices, and the second is the fog layer. The following is a comprehensive explanation of the suggested system:

3.1 IoT layer

This layer consists of two Raspberry Pi devices and several sensors. The temperature sensor on the first device and the temperature and humidity sensor on the second have unique IP addresses (192.168. 0.9 and 192.168. 0.10). These devices use the Python programming language and the TCP protocol, port

2024, to send data to a fog computing server with an address of 192.168.0.200 fog computing.

The IoT layer's compact and low-power sensor nodes continuously gather environmental data. These data are sent securely to the fog layer, where offline-trained models ensure that even resource-limited IoT devices benefit from robust anomaly detection without incurring additional computational overhead locally.

a) Reading sensor data

The gathering technique involves using two Raspberry Pi devices: a Raspberry Pi 4 Model B with 4 GB of RAM and a Raspberry Pi 3 Model A+ with an 8 GB microSD card slot. The devices included in Table 1 are a temperature sensor and a temperature and humidity sensor to read sensor data. These devices are programmable computers facilitating communication and supporting various network protocols and peripherals. This is made possible due to their compact size and affordable cost. Due to its compact size and affordable price, the Raspberry Pi is a competent and efficient computer.

Table 1. Specifications of device nodes

Device	Model	Size	Voltage
1 Raspberry Pi	4 Model B	85.6mm × 56.5mm × 17mm	5V(3A)
1 Raspberry Pi	3 Model A+	65mm × 56mm × 12mm	5(2.5A)
2 Temperature Sensor	DS18B20 Waterproof Digital	6*50mm	3 to 5.5 V
2 Temperatureand humidity sensor	DHT11 Digital	3cm*1cm	3.5V to 5.5V

b) Authentication and encryption

In a fog computing environment, when a sensor for humidity and temperature senses communication, each party needs to get authenticated, exchange public keys, and generate a shared key using the ECDH algorithm. Following the two parties' authentication to create a shared key, the data will be encrypted using the Speck algorithm and decrypted in the server, as can be seen in Figure 1 below; authentication was completed, a communication channel was established, and a shared key was decided upon for data encryption in the IoT environment (represented by the Raspberry Pi) and decryption in the fog computing environment (represented by the server computer). The humidity and temperature were measured at 28.5 degrees Celsius.

```
Connection from ('192.168.0.10', 60578) has been established.
Decrypted Message: 28.375
Connection from ('192.168.0.10', 60580) has been established.
Decrypted Message: 28.5
Connection from ('192.168.0.10', 60582) has been established.
Decrypted Message: 28.437
Connection from ('192.168.0.10', 60584) has been established.
Decrypted Message: 28.5
Connection from ('192.168.0.10', 60586) has been established.
Decrypted Message: 28.5
Connection from ('192.168.0.10', 60588) has been established.
```

Figure 1. Decryption process in fog node

Speck was selected for its lightweight properties, making it ideal for constrained IoT devices. By coupling Speck encryption with ECDH key exchange, the system ensures end-to-end data confidentiality and integrity, allowing secure real-time communications. This combination provides a balanced security model that supports the low-latency requirements of IoT environments.

Algorithm 1 shows the main steps of the Speck lightweight encryption stage

Algorithm 1. Pseudo code for speck algorithm (Encryption stage)

```
Input: plaintext p, encryption key K
Output: cipher text C
1: Split plaintext p into two n-bit values: x and y
     x ← p[0: n-1]
     y ← p[n:2n-1]
2: Initialize constants
     T ← number_of_rounds
     α ← some_ constant
     β ← some_constant
3: Generate round keys
     Generate round keys K_0, K_1, ..., K_(T-1) using the key K
4: For i = 0 to T-1 do
     4.1: Update x
       x ← (x + y) ⊕ (S^(-α) (x + y)) ⊕ K_i
     4.2: Update y
       y ← (S^β (y)) ⊕ x
     End For Loop
5: Combine the final values of x and y to get the cipher text C
    C ← combine (x, y)
End Algorithm
```

3.2 Fog layer

The work was done from a computer with Core i7 features and 256 GB SSD RAM. The sensors used are based on detecting the extent of changes that occur in the surrounding environment and transferring the information obtained to the fog layer in the network. In the online phase, the fog layer uses the pre-trained model (from the offline phase) to analyze incoming traffic in real-time. By converting raw packet data into entropy and Chi-Square values over defined sliding windows, the fog layer promptly classifies traffic as usual or suspicious, enabling immediate, informed responses to potential DDoS attacks.

a) Set (n) window time

It is essential in contributing to real-time data retrieval. Several applications, such as Wireshark, support this stage and are powerful tools for analyzing traffic networks. This inquiry will utilize Wireshark, a notable software tool for analyzing network traffic. This software is both open-source and free to use, and it can be easily installed on machines running the Windows operating system. During packet analysis, Wireshark examines four important features of the packet in the network: the source address, the source port, the destination address, and the destination port, to help reduce and identify any attacks that occur during communication between nodes. The data will be received at regular intervals of exactly (20) seconds until it has been comprehensively analyzed.

b) Convert window time to a numeric value

Measurement of the network's randomness using entropy and Chi-Square for four attributes (Src_IP, Port_IP, Dst_IP, and Dst_Port) is required to identify distributed service attacks. These properties were transformed into values using the Claude Shannon-proposed mathematical equation for entropy, the standard entropy formula in Eq. (1) [27]. a measure of predictive power, and the Chi-Square test, a statistical method, to ascertain the level of subjective reliability between the two property formulas found in Eq. (2) [28]. It determines whether

the observed and anticipated values of the equation deviate significantly from one another. After it was transformed into values, the Chi-Square and entropy values were added together. Every (100) packets are collected using Python, the Pyshark package, and Wireshark. Each receiver package's four attributes (source IP/port, destination IP/port, and package) are obtained and entered into the sliding window period.

$$H(x) = -\sum_{i=1}^{n} p(xi) log_2\big(p(xi)\big) \quad (1)$$

where, H is entropy, and P is Probability.

$$\sum x^2{}_{i-j=\frac{(O-E)^2}{E}} \quad (2)$$

where, c=degree of freedom, O=observed values, E=Expected Values.

Combining entropy and Chi-Square statistics enhances the model's sensitivity to subtle deviations in traffic behavior. Entropy quantifies randomness and highlights unusual patterns, while the Chi-Square test assesses how observed distributions differ from expectations. Their joint use provides a more robust feature representation, improving the classifier's precision in distinguishing between benign and malicious traffic.

Algorithm 2 shows the main steps of entropy and Chi-Square sliding windows calculation.

Algorithm 2. Calculate the entropy& Chi-Square Sliding Window

Input: Sliding window (Src_IP, Port_IP, Dst_IP, and Dst_Port)
Output: Chi-Square value (χ^2), Entropy (H)
1: Initialize sliding_window
While (every incoming packet) do
2: Filter out packets with null values
    If packet is null, then
      Pass
    End If
3: Extract relevant features from the packet:
    source_ip ← packet.Source_IP
    source_port ← packet.Source_Port
    destination_ip ← packet.Destination_IP
    destination_port ← packet.Destination_Port
4: Add extracted features to the sliding window
    sliding_window. append((Src_IP, Port_IP, Dst_IP, and Dst_Port))
5: If sliding_window is full (reaches max size), then
  6: Calculate Entropy
    Call Entropy(sliding_window)
  7: Calculate Chi-Square
    Call Chi_Square(sliding_window)
  8: Clear or shift the sliding window for the next set of 100 packets
  End If

4. METHODOLOGY

This methodology integrates two primary stages: (1) an offline preprocessing and training stage, where datasets are prepared, balanced with SMOTE, and enhanced using Entropy and Chi-Square feature analyses; and (2) an online deployment stage, where the trained model is applied within a fog computing environment. The following subsections detail the datasets, feature extraction techniques, handling of class imbalance, the selection and training of classifiers, and the evaluation metrics used to measure performance and real-time responsiveness. The proposed methodology consists of many preprocessing steps in the following sub-sections.

4.1 Classification methods

We used two publicly accessible datasets: the Canadian Cybersecurity Institute's (CIC) CIC-DDoS2019 [29] dataset. Its entire number of records is 50,063,112, according to reference [30]. The CSV files make up the datasets. UDP, SSDP, SYN, NTP, NETBOIS, MSSQL, LDAP, and udp_lag is captured in the CIC-DDoS2019 dataset concerning training and testing days. The quantity of benign traffic in this dataset is small compared to the abundant aggressive traffic. This presents a hurdle because innocuous occurrences are not sufficiently represented during training. As a result, we considered most of the innocuous traffic that was captured from a single file taken during each attack.

The Edge-IIoTset [31] dataset used in this study is publicly accessible on Kaggle. It includes records for 14 types of attacks and five categories of attacks in the original dataset. However, since our research focuses solely on DDoS attacks, we eliminate all other attack types. Consequently, we are left with data containing only DDoS attacks and average traffic data. The dataset names TCP flood attacks, SYN flood attacks, UDP flood attacks, HTTP flood attacks, and ICMP flood attacks as the other four DDoS attacks. The data utilized is separated into two categories: the threat category and the normal category. These datasets were chosen due to their comprehensive coverage of modern DDoS attack vectors and realistic traffic patterns. CIC-DDoS2019 offers a broad range of attack types with highly imbalanced classes, while Edge-IIoTset reflects heterogeneous IoT scenarios. Using both datasets ensure the trained models can generalize across diverse conditions.

Four attacks were taken from the CIC-DDoS2019 and Edge-IIoT datasets, two attacks from each group, and the entropy and Chi-Square values were calculated. The results were then gathered and obtained a third time, and they are used in the proposed model as indicated in Table 2. Before training by 80% and testing by 20% on the SVM, DT, and RF algorithms, the results are finally balanced using SMOTE, which aims to equalize the distribution of classes by creating artificial instances for the minority class. By using extrapolation techniques to estimate values between existing examples of the minority class and their nearest neighbors, new training records are generated. SMOTE helps address the issue of overfitting caused by random oversampling by selectively oversampling the minority class [32].

To address class imbalance, we applied SMOTE with carefully tuned parameters to synthesize new minority-class samples. This approach reduces bias towards majority classes, ensuring that the classifier learns robust decision boundaries. Preliminary experiments confirmed that SMOTE improved recall for underrepresented attack types without overfitting.

We have identified and extracted four key features: source IP, source port, destination IP, and destination port. Three machine learning approaches (ML) will be used in the proposed model to train on these datasets: RF and SVM.

Table 2. Data distribution utilized in the suggested model

Dataset Used	Attack	No. of DDoS	No. of Benign	Entropy	Chi-Square	Entropy& Chi-Square
CIC-DDoS2019	SYN	1048228	347	10701	10701	10701
	UDP	1047441	1134	10693	10693	10693
Edge-IIoTset	http	10561	24301	349	349	349
	TCP	10247	24301	346	346	346

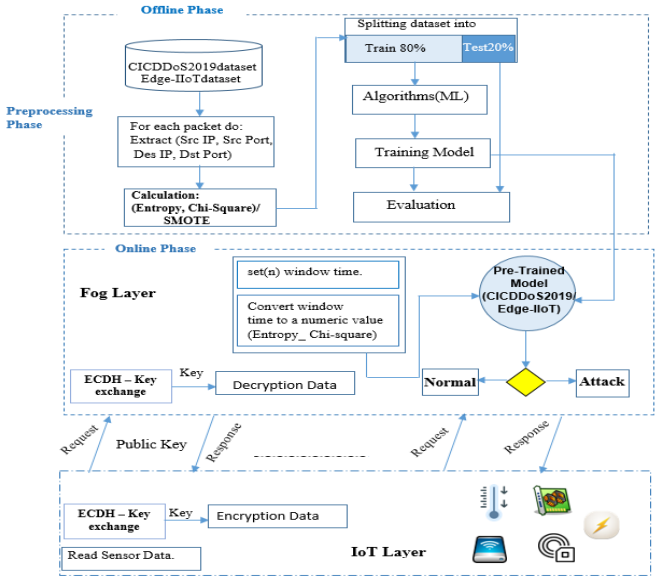

Figure 2. Flowchart of the proposed system

SVM was chosen because it is able to handle highly dimensional data robustly and contributes to finding solutions that support optimal decisions during classification. DT provides interpretability and efficient handling of both numerical and categorical features. RF, as an ensemble of DT, offers robustness against noise and improves generalization. Collectively, these algorithms are well-suited to detecting complex DDoS patterns in IoT traffic and can be efficiently implemented at the fog layer (Figure 2).

a) Support vector machine (SVM)

The support-vector network's concept is to transform input vectors into a higher-dimensional feature space, Z, using a predetermined non-linear mapping.

The unique properties of an established linear decision boundary in this field enhance the network's capacity for efficient information generalization. Each data point in SVM is represented as a point in a multidimensional space, with dimensions that match the characteristics of the data. The approach visualizes the data in this space and searches for an optimal hyperplane that partitions the two groups. The SVM categorizes new instances based on them location in the hyperplane, which acts as the decision boundary [33].

b) Random forest classifier (RF)

One technique for categorization that involves creating numerous classification trees is RF and DT are classification trees that are employed in RF. The number of classification trees created influences the accuracy of the categorization outcomes. The range of 64 to 128 DT is ideal Every decision tree in the RF classification makes a forecast, and the RF final prediction is determined by the majority vote of the trees' predictions [34].

c) Decision tree classifier (DT)

It is considered one of the supervised learning techniques that contribute to the analysis of the regression of packets and their classification in the network. The DT performs best when dealing with both numerical and category data. To forecast target variables, these hierarchical data structures partition the input data space into multiple subspaces [35, 36].

Algorithm 3 shows the main machine-learning pseudo-code.

Algorithm3: Machine learning algorithms pseudo-code

Input: - Features: (Src_IP, Port_IP, Dst_IP, and Dst_Port)
- Train data: CIC-DDoS2019, Edge-IIoTset
- Classifiers: SVM, DT, RF
Output: (Class label: Attack or Normal)
Begin
1: Load the training data - Train data ← load_data(CIC-DDoS2019, Edge-IIoTset)
2: Pre-process the training data - Extract features: (Src_IP, Src_ Port, Dst_IP, Dst_Port)
- Extract class labels: Attack or Normal
3: Train classifiers (SVM, DT, RF)
4: Test phase using sliding window
While (incoming test data) do
5: Collect test data into the sliding window - Extract features from each incoming packet:
source_ip ← packet.Source_IP
source_port ← packet.Source_Port
destination_ip ← packet.Destination_IP
destination_port ← packet.Destination_Port
- Add extracted features to the sliding window
6: Calculate Entropy for the sliding window
- entropy_value ← calculate_entropy(sliding_window)
7: Calculate the Chi-Square value for the sliding window
-chi_square_value← calculate_chi_square(sliding_window)
8: Create feature vector for classification
- feature_vector ← [entropy_value, chi_square_value] Step 9: Predict class label using the trained classifiers (SVM, RF, DT)
10: Label sliding window
If final_prediction = "Attack" then
Label sliding window as "Attack"
Else: Label sliding window as "Normal"
End IF and While
End

4.2 Evaluation metrics

The following measures are used to assess the model's performance: The model is evaluated using the evaluation metrics are as follows:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

4.2.1 Precision

It is the ratio of correctly predicted attack results to the network's total number of attack categories.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{4}$$

4.2.2 Recall

It is used to evaluate the effectiveness of a classification model to identify DDoS attacks by measuring the ratio of correctly classified positive cases to the total number of

positive cases that attack the network, calculated as follows:

$$\text{Recall} = \frac{TP}{TP + FN} \quad (5)$$

4.2.3 F1-Score

It is used to calculate the harmonic mean balanced between precision and recall, especially when the data is distributed unevenly, as is the case in DDoS attacks, as it contributes to evaluating the effectiveness of the classification model to identify true positive cases and reduce false positive ones. The Eq. (6) is:

$$\text{F1} - \text{Score} = 2.\frac{\text{Precision.Recall}}{\text{Precision} + \text{Recall}} \quad (6)$$

4.2.4 Throughput

It is used to evaluate the download efficiency of completed packets within a specified period. The high value contributes to increased throughput and improved download efficiency while classifying DDoS attacks [37].

$$\text{Throughput} = \left(\frac{\text{Total Num of send and received packet}}{\text{Time}}\right) \quad (7)$$

4.2.5 Response time

It is the takes for the system to respond during an attack. It is considered very important to assess the impact of the attack to respond to the actual requests of operations in the network, which gives an impression of the availability and performance of the service. This revealed that the result is dependent on the Mt and Et. [38]:

$$\text{Rt} = \text{Mt} + \text{Et} + \text{NdL}. \quad (8)$$

4.2.6 Execution time

As stated in Eq. (9), the performance of a computer operating system depends on various elements, such as the number of processors (Js) and the speed of the central processing unit (CPUs) [39].

$$E_{t=\frac{J_s}{CPU_t}} \quad (9)$$

4.2.7 Latency

The phrase refers to the time interval between when the load balancer gets a request and when it sends a response [40].

$$d_{trans} = {}^{L}/_{R} \quad (10)$$

The components of the equation are represented by the letter D, which represents the packet delay time in seconds, the letter L represents the length of the transmitted packet in bits, and the letter R represents the rate of data transmitted between nodes in bits during each time unit [40].

5. RESULTS AND DISCUSSION

The SVM, DT, and RF algorithms were tested on both CIC-DDoS2019 and Edge-IIoTset datasets.

The accuracy, precision, recall, and F1-score metrics were computed. SVM achieved 99% accuracy for the SYN attack, while DT and RF performed similarly with 98% and 99% respectively. On the HTTP attack, RF outperformed SVM and DT, reaching 94% accuracy.

These performance differences can be attributed to the inherent characteristics of each classifier and the preprocessing techniques applied. SVM's ability to find an optimal decision boundary in high-dimensional spaces often leads to strong performance on complex attack patterns. DT's interpretability and efficient handling of both numerical and categorical features allow it to adapt quickly to varying traffic distributions. RF, benefiting from ensemble voting and reduced variance, demonstrates robustness against noisy or imbalanced data. The combined use of Entropy and Chi-Square ensured that only highly discriminative features were used, enhancing classifier sensitivity. Moreover, the application of SMOTE to balance minority classes contributed to improved recall rates, ensuring that subtle attack signatures were not overshadowed by majority classes. Collectively, these factors explain why certain classifiers excelled in detecting specific DDoS attacks, ultimately leading to more reliable and generalizable detection performance. Subsequently, we evaluated network parameters such as latency and throughput to ensure that the detection system operates effectively in real-time environments.

This section summarizes the results of the proposed system, which consists of two stages: offline and online.

5.1 Detection attack in offline

In this point, the offline results of a trained model for detecting DDoS attacks are presented in this stage. Two attacks, namely the SYN flood attack and the UDP flood attack, were selected from the CIC-DDoS2019 dataset. Then, the data were processed using an entropy equation; entropy results were obtained in Figure 3, and Chi-Square equations and Chi-Square results were obtained in Figure 4. Next, we combine the findings from the Chi-Square and entropy values to get good results, as shown in Figure 5. Adopted in the proposed model. Three algorithms, namely SVM, RF Classifier, and DT, were applied to each attack.

Table 3. Results of SVM, DR, and RF accuracy using entropy for CIC-DDoS2019 dataset

Classifier	Attack	Accuracy	Recall	Precision	F1-Score
SVM		0.99	0.99	0.99	0.99
DT	SYN	0.99	0.99	0.99	0.99
RF		0.99	0.99	0.99	0.99
SVM		0.98	0.98	0.98	0.98
DT	UDP	0.98	0.98	0.98	0.98
RF		0.99	0.99	0.99	0.99

The entropy findings for the SYN attack are displayed in Table 3. The three algorithms—SVM, DT, and RF—obtained a precision of 0.99 in this case. They also achieved an accuracy of 0.99 in the remaining metrics, which include recall, F1-score, and precision. The SVM and DT techniques obtained an accuracy of 0.98 for the UDP attack, while the RF algorithm reached 0.99 for the remaining metrics. The RF algorithm also achieved an accuracy 0.99 for the remaining recall metrics, f1-score, and precision.

Table 4 displays the Chi-Square results for the Syn flood attack. The SVM algorithm demonstrated an accuracy of 0.99 while also achieving 0.99 in the precision, recall, and f1-score metrics. The DT and RF algorithms showed an accuracy of

0.98 in the precision, recall, and f1-score metrics. The SVM and DT algorithms performed 0.98 well in the UDP flood attack and 0.98 well in the precision, recall, and f1-score metrics. The RF method performed 0.99 well in the UDP flood attack; the accuracy reached 0.99 in the precision, recall, and f1-score metrics.

Table 4. Accuracy results of CIC-DDoS2019Set using Chi-Square

Classifier	Attack	Accuracy	Recall	Precision	F1-Score
SVM		0.99	0.99	0.99	0.99
DT	SYN	0.98	0.98	0.98	0.98
RF		0.98	0.98	0.98	0.98
SVM	UDP	0.98	0.98	0.98	0.98
DT		0.98	0.98	0.98	0.98

Table 5. Accuracy results for CIC-DDoS2019 dataset using calculation of entropy and Chi-Square

Classifier	Attack	Accuracy	Recall	Precision	F1-Score
SVM		1.00	1.00	1.00	1.00
DT	**SYN**	1.00	1.00	1.00	1.00
RF		1.00	1.00	1.00	1.00
SVM		1.00	1.00	1.00	1.00
DT	**UDP**	1.00	1.00	1.00	1.00
RF		1.00	1.00	1.00	1.00

The results in Table 5, which displays the outcomes of applying three algorithms to each attack are presented. The precision, recall, accuracy, and f1-score criteria were used to evaluate these methods. The SVM, RF, and DT algorithms all had a 100% success rate in detecting the SYN Flood attack. Similarly, all three algorithms used in the UDP attack demonstrated perfect scores of 100 for precision, recall, accuracy, and f1-score metrics. The efficiency of the model trained on the CIC-DDoS2019 dataset enables it to detect DDoS attacks accurately.

Two attacks—an HTTP attack and a TCP attack—were selected from the Edge-IIoTset dataset. Then, the entropy and Chi-Square values of the data were calculated in Figures 3 and 4.

The HTTP attack entropy findings are displayed in Table 6. The accuracy of the SVM and DT algorithms was 0.898. Precision was 0.899, and recall was 0.90; the f1-score was 0.898. The RF algorithm's accuracy was 0.949. They also attained 0.949 precision in the remaining metrics, recall, and f1-score. Three algorithms—SVM, DT, and RF—achieved 0.99 precision in the TCP attack. They obtained a 0.99 in the metrics of recall, precision, and F1-score.

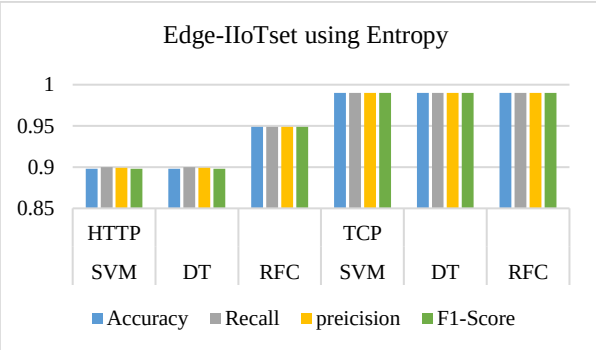

Figure 3. Accuracy results of machine learning (SVM, DR, and RF) using entropy for HTTP and TCP attacks

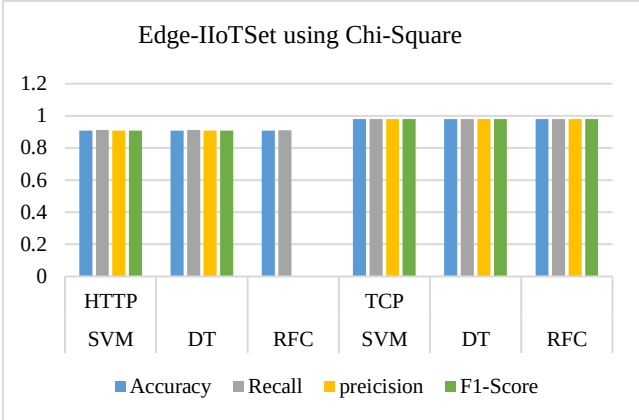

Figure 4. Accuracy results of machine learning (SVM, DR, and RF) using Chi-Square

Table 6. Machine learning accuracy result for Edge-IIoTset using entropy

Classifier	Attack	Accuracy	Recall	precision	F1-Score
SVM		0.898	0.90	0.899	0.898
DT	HTTP	0.898	0.90	0.899	0.898
RF		0.949	0.949	0.949	0.949
SVM		0.99	0.99	0.99	0.99
DT	TCP	0.99	0.99	0.99	0.99
RF		0.99	0.99	0.99	0.99

Figure 4 shows the accuracy of two attacks, HTTP and TCP, for all machine learning algorithms (SVM, DT, and RF) using the Chi-Square.

Table 7 displays the Chi-Square findings for the HTTP attack in the SVM and DT algorithms. Both methods achieved an accuracy of 0.909, recall of 0.913, precision of 0.909, and f1-score of 0.908. The RF algorithm produced the following results: recall of 0.91, accuracy of 0.909, precision of 0.909, and f1-score of 0.908. The SVM, DT, and RF algorithms all obtained an accuracy of 0.98 in the TCP attack. They both scored 0.98 in the remaining metrics: F1-score, recall, and precision.

Table 7. Machine learning accuracy results for Edge-IIoTset using Chi-Square

Classifier	Attack	Accuracy	Recall	precision	F1-Score
SVM		0.909	0.913	0.909	0.908
DT	http	0.909	0.913	0.909	0.908
RF		0.909	0.91	0.909	0.908
SVM		0.98	0.98	0.98	0.98
DT	TCP	0.98	0.98	0.98	0.98
RF		0.98	0.98	0.98	0.98

The three algorithms, SVM, DT, and RF, were applied for each attack, and the entropy and Chi-Square values were combined, as shown in Figure 5, to produce good and trustworthy results in the proposed model.

Table 8 below shows three classifiers, one for each attack. Each classifier assesses a different set of metrics. The SVM classifier in the HTTP attack achieved precision, F1-Score, accuracy, and recall of 0.93 and 0.94, respectively. The DT classifier's accuracy, precision, F1-Score, and recall were 0.93, 0.94, and 0.94, respectively.

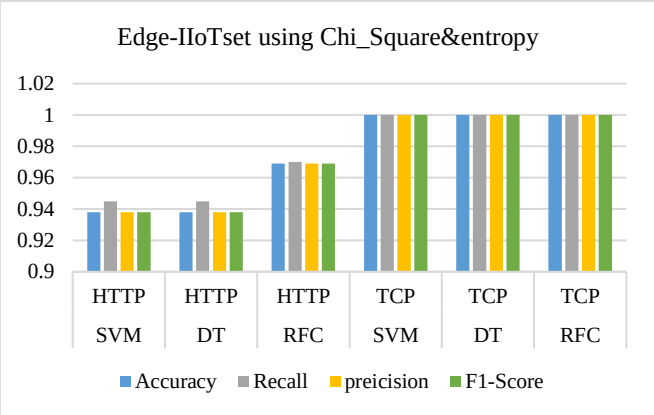

Figure 5. Accuracy results of machine learning (SVM, DR, and RF) using Chi-Square & entropy

Regarding the RF classifier, it attained precision, accuracy, F1-Score, and recall of 0.97, 0.96, and 0.96, respectively. We observe that in terms of identifying the HTTP attack, the RF classifier performs better than the other classifiers. Three classifiers detected the TCP attack with an accuracy of 100. Likewise, accuracy was attained in the remaining measures. It is observed that the model. It performs better in identifying TCP attacks than in identifying HTTP attacks.

Table 8. Machine learning accuracy results in Edge-IIoTset by entropy & Chi-Square

Classifier	Attack	Accuracy	Recall	Precision	F1-Score
SVM		0.938	0.945	0.938	0.938
DT	**HTTP**	0.938	0.945	0.938	0.938
RF		0.969	0.97	0.969	0.969
SVM		1.00	1.00	1.00	1.00
DT	**TCP**	1.00	1.00	1.00	1.00
RF		1.00	1.00	1.00	1.00

5.2 Detection attack in real-time

At this point, the pre-trained model—which comprises the three algorithms SVM, DT, and RF—is used to detect whether the network condition is normal or under attack. That is the case when two algorithms concur on whether a network is under attack. The metrics used to test the network performance during authentication, encryption, and decryption were throughput, which was recorded at 0.991k/sec; response time, which was reported at 0.004sec; latency, which was recorded at 0.015sec; and execution time, which was recorded at 1.033sec. When the Wireshark program detects a DDoS attack in real-time, see Figure 6.

Table 9 shows the proposed work comparisons with other related works to detect DDoS attacks. Table 9 clearly shows that the suggested model detected DDoS attacks with a higher degree of accuracy than earlier efforts.

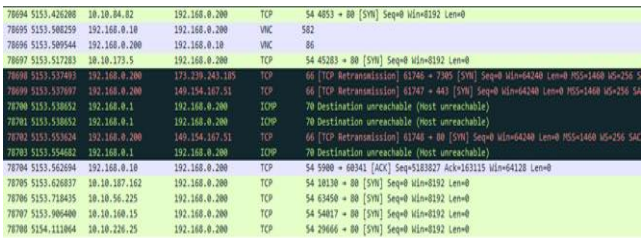

Figure 6. DDoS attacks in real-time in Wireshark program

Table 9. A comparison between the suggested model and related work for machine learning algorithms

Ref.	Year	Classification	Dataset	Real-Time	Accuracy
[14]	2018	SVM	predefined parameters	Yes	**0.97**
[15]	2018	SVM	a limited feature set	Yes	**0.99**
[16]	2021	K nearest neighbor, DT NB	CIC-DDoS 2019	Yes	**99.99%** **99.88%** **94.55%**
[17]	2021	Random Forest	CIC-DDoS2019	No	**99.92%**
[19]	2021	SVM K nearest neighbor, RF	CIC-DDoS2019	No	**99.77%** **99.77%** **96.36%**
[23]	2022	Inception Time algorithm	Edge-IIoTset	No	**94.36%**
[24]	2023	RF	Edge-IIoTset	No	**94%**
[25]	2023	XGBoost	Edge-IIoTset	No	**99.88%**
The proposed system		**SVM** **RF** **DT**	**CIC-DDoS2019**	**Yes**	**100** **100** **100**
		SVM **RF** **DT**	**Edge-IIoTset**	**Yes**	**100** **96%** **100**

6. CONCLUSIONS

To quickly identify DDoS attacks, we aimed to establish a safe fog computing environment in this paper. We used SVM, DT, and RF machine learning algorithms in the first offline stage to design a pre-trained model. We then trained the model on the CIC-DDoS2019 and Edge-IIoTset dataset, extractedfour features (SreIP, Port_IP, Dst_IP, and Dst_Port), calculated the Chi-Square entropy, and tested the model using TCP, UDP, and SYN attacks. Except for HTTP, all algorithms yielded a 100-accuracy rate. In the DT and SVM algorithms, it obtained an accuracy of 0.93; in the RF method, it received an accuracy of 0.96. During the online phase, the model was evaluated in real-time by monitoring network latency, encrypting data using the Speck method, and verifying communication between transmitting and receiving parties. performance, a collection of parameters including latency, response time, execution time, throughput, and the ability to use the learned model to identify attacks in real time. When a DDoS assault is detected in the future, we will employ the quantum technique, close the port, and train the model using data that includes all kinds of DDoS attacks.

REFERENCES

[1] Al-Mashhadani, M., Shujaa, M. (2022). IoT security using AES encryption technology based ESP32 platform. The International Arab Journal of Information Technology, 19(2): 214-223. https://doi.org/10.34028/iajit/19/2/8

[2] Hassan, K.F., Manaa, M.E. (2022). Detection and mitigation of DDoS attacks in Internet of Things using a fog computing hybrid approach. Bulletin of Electrical Engineering and Informatics, 11(3): 1604-1613. https://doi.org/10.11591/eei.v11i3.3643

[3] Saad, Z.M., Mhmood, M.R. (2023). Fog computing system for Internet of Things: Survey. Texas Journal of Engineering and Technology, 16: 1-10. https://zienjournals.com/index.php/tjet/article/view/3163.

[4] Atlam, H.F., Walters, R.J., Wills, G.B. (2018). Fog computing and the Internet of Things: A review. Big Data and Cognitive Computing, 2(2): 10. https://doi.org/10.3390/bdcc2020010

[5] Weng, C.Y., Li, C.T., Chen, C.L., Lee, C.C., Deng, Y.Y. (2021). A lightweight anonymous authentication and secure communication scheme for fog computing services. IEEE Access, 9: 145522-145537. https://doi.org/10.1109/ACCESS.2021.3123234

[6] Zhou, L., Guo, H., Deng, G. (2019). A fog computing based approach to DDoS mitigation in IIoT systems. Computers & Security, 85: 51-62. https://doi.org/10.3390/en14154676

[7] Li, X.L., Chen, Z.B. (2022). DDoS attack detection by hybrid deep learning methodologies. Security and Communication Networks, 2022(1): 7866096. https://doi.org/10.1155/2023/9810961

[8] Najafimehr, M., Zarifzadeh, S., Mostafavi, S. (2022). A hybrid machine learning approach for detecting unprecedented DDoS attacks. The Journal of Supercomputing, 78(6): 8106-8136. https://doi.org/10.1007/s11227-021-04253-x

[9] Saleh, M., Jhanjhi, N.Z., Abdullah, A., Saher, R. (2022). Proposing encryption selection model for IoT devices based on IoT device design. In 2022 24th International Conference on Advanced Communication Technology (ICACT), pp. 210-219. https://doi.org/10.23919/ICACT53585.2022.9728914

[10] Di Matteo, S., Baldanzi, L., Crocetti, L., Nannipieri, P., Fanucci, L., Saponara, S. (2021). Secure elliptic curve crypto-processor for real-time IoT applications. Energies, 14(15): 4676. https://doi.org/10.3390/en14154676

[11] Soni, N.K., Patel, T.P. (2014). Quality teaching & higher education system in India. International Journal of Scientific and Research Publications, 4(1).

[12] Li, N. (2010). Research on Diffie-Hellman key exchange protocol. In 2010 2nd International Conference on Computer Engineering and Technology, Chengdu, China, pp. V4-634-V4-637. https://doi.org/10.1109/ICCET.2010.5485276

[13] AbdulRaheem, M., Balogun, G.B., Abiodun, M.K., Taofeek-Ibrahim, F.A., Tomori, A.R., Oladipo, I.D., Awotunde, J.B. (2021). An enhanced lightweight speck system for cloud-based smart healthcare. In Applied Informatics: Fourth International Conference, ICAI 2021, Buenos Aires, Argentina, pp. 363-376. https://doi.org/10.1007/978-3-030-89654-6_26

[14] Al-Razaq, F.J.A., Mohammed, S.J., Manaa, M.E., Al-Murieb, S.S.A., Al-Khamees, H.A.A. (2024). Classification model of spam emails based on data mining – deep learning techniques. International Journal of Safety and Security Engineering, 14(4): 1195-1202. https://doi.org/10.18280/ijsse.140416

[15] Mallampati, D., Hegde, N.P. (2024). Enhanced detection of text and image spam using cost-sensitive deep learning. Traitement du Signal, 41(3): 1283-1292. https://doi.org/10.18280/ts.410317

[16] Lawal, M.A., Shaikh, R.A., Hassan, S.R. (2021). A DDoS attack mitigation framework for IoT networks using fog computing. Procedia Computer Science, 182: 13-20. https://doi.org/10.1016/j.procs.2021.02.003

[17] Singh Samom, P., Taggu, A. (2021). Distributed denial of service (DDoS) attacks detection: A machine learning approach. In Applied Soft Computing and Communication Networks: Proceedings of ACN 2020, pp. 75-87. https://doi.org/10.1007/978-981-33-6173-7_6

[18] Khoei, T.T., Aissou, G., Hu, W.C., Kaabouch, N. (2021). Ensemble learning methods for anomaly intrusion detection system in smart grid. In 2021 IEEE International Conference on Electro Information Technology (EIT), Mt. Pleasant, MI, USA, pp. 129-135. https://doi.org/10.1109/EIT51626.2021.9491891

[19] Yungaicela-Naula, N.M., Vargas-Rosales, C., Perez-Diaz, J.A. (2021). SDN-based architecture for transport and application layer DDoS attack detection by using machine and deep learning. IEEE Access, 9: 108495-108512. https://doi.org/10.1109/ACCESS.2021.3101650

[20] Mishra, A. (2022). Prediction approach against DDoS attack based on machine learning multiclassfier. arXiv preprint arXiv:2204.12855. https://doi.org/10.48550/arXiv.2204.12855

[21] Ogini, N.O., Adigwe, W., Ogwara, N.O. (2022). Distributed denial of service attack detection and prevention model for IoT-based computing environment using ensemble machine learning approach. International Journal of Network Security & Its Applications, 14(4): 39-53. https://doi.org/10.5121/ijnsa.2022.14403

[22] Garg, S., Kumar, V., Payyavula, S.R. (2022). Identification of Internet of Things (IoT) attacks using gradient boosting: A cross dataset approach. Telematique, 21(1): 6982-7012.

[23] Tareq, I., Elbagoury, B.M., El-Regaily, S., El-Horbaty, E.S.M. (2022). Analysis of ton-IoT, unw-nb15, and edge-IIoT datasets using dl in cybersecurity for IoT. Applied Sciences, 12(19): 9572. https://doi.org/10.3390/app12199572

[24] Hamza, N., Lakmal, H.K.I.S., Maduranga, M.W.P., Kathriarachchi, R.P.S. (2023). Malware detection of IoT networks using machine learning: An experimental study with edge IIoT dataset. In 30th Annual Technical Conference-IET Sri Lanka Network, Colombo, Sri Lanka.

[25] Laiq, F., Al-Obeidat, F., Amin, A., Moreira, F. (2023). DDoS attack detection in edge-IIoT using ensemble learning. In 2023 7th Cyber Security in Networking Conference (CSNet), pp. 204-207. https://doi.org/10.1109/CSNet59123.2023.10339784

[26] Rashid, M.M., Khan, S.U., Eusufzai, F., Redwan, M.A., Sabuj, S.R., Elsharief, M. (2023). A federated learning-based approach for improving intrusion detection in industrial Internet of Things networks. Network, 3(1): 158-179. https://doi.org/10.3390/network3010008

[27] Behal, S., Kumar, K. (2017). Detection of DDoS attacks and flash events using novel information theory metrics. Computer Networks, 116: 96-110. https://doi.org/10.1016/j.comnet.2017.02.015

[28] Thaseen, I.S., Kumar, C.A. (2017). Intrusion detection model using fusion of Chi-Square feature selection and multi class SVM. Journal of King Saud University-Computer and Information Sciences, 29(4): 462-472. https://doi.org/10.1016/j.jksuci.2015.12.004

[29] Sharafaldin, I., Lashkari, A.H., Hakak, S., Ghorbani, A.A. (2019). Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy. In 2019 International Carnahan Conference on Security Technology (ICCST), pp. 1-8. https://doi.org/10.1109/CCST.2019.8888419

[30] Ferrag, M.A., Shu, L., Djallel, H., Choo, K.K.R. (2021). Deep learning-based intrusion detection for distributed denial of service attack in agriculture 4.0. Electronics, 10(11): 1257. https://doi.org/10.3390/electronics10111257

[31] Ferrag, M.A., Friha, O., Hamouda, D., Maglaras, L., Janicke, H. (2022). Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning. IEEE Access, 10: 40281-40306. https://doi.org/10.1109/ACCESS.2022.3165809

[32] Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P. (2002). SMOTE: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research, 16: 321-357. https://doi.org/10.1613/jair.953

[33] Cervantes, J., Garcia-Lamont, F., Rodríguez-Mazahua, L., Lopez, A. (2020). A comprehensive survey on support vector machine classification: Applications, challenges and trends. Neurocomputing, 408: 189-215. https://doi.org/10.1016/j.neucom.2019.10.118

[34] Resende, P.A.A., Drummond, A.C. (2018). A survey of random forest based methods for intrusion detection systems. ACM Computing Surveys (CSUR), 51(3): 1-36. https://doi.org/10.1016/j.cose.2024.104209

[35] Jin, C., Li, F., Ma, S., Wang, Y. (2022). Sampling scheme-based classification rule mining method using decision tree in big data environment. Knowledge-Based Systems, 244: 108522. https://doi.org/10.1016/j.knosys.2022.108522

[36] Javeed, D., Gao, T., Saeed, M.S., Kumar, P. (2023). An intrusion detection system for edge-envisioned smart agriculture in extreme environment. IEEE Internet of Things Journal, 2023: 3288544. https://doi.org/10.1109/JIOT.2023.3288544

[37] Okwu, M.O., Tartibu, L.K. (2021). Particle swarm optimisation. Metaheuristic Optimization: Nature-Inspired Algorithms Swarm and Computational Intelligence, Theory and Applications, pp. 5-13. https://doi.org/10.1007/978-3-030-61111-8_2

[38] Radhika, E.G., Sadasivam, G.S. (2021). A review on prediction based autoscaling techniques for heterogeneous applications in cloud environment. Materials Today: Proceedings, 45: 2793-2800. https://doi.org/10.1016/j.matpr.2020.11.789

[39] Desebrock, C., Spence, C. (2021). The self-prioritization effect: Self-referential processing in movement highlights modulation at multiple stages. Attention, Perception, & Psychophysics, 83(6): 2656-2674. https://doi.org/10.3758/s13414-021-02295-0

[40] Chalapathi, G.S.S., Chamola, V., Vaish, A., Buyya, R. (2021). Industrial Internet of Things (IoT) applications of edge and fog computing: A review and future directions. Fog/Edge Computing for Security, Privacy, and Applications, 293-325. https://doi.org/10.1007/978-3-030-57328-7_12