Computer Modeling in Engineering & Sciences

Tech Science Press

Doi:10.32604/cmes.2025.068326

ARTICLE

A Hybrid Machine Learning and Blockchain Framework for IoT DDoS Mitigation

Singamaneni Krishnapriya[1,2,*] **and Sukhvinder Singh**[1]

[1]Department of Computer Science, School of Engineering and Technology Pondicherry University, Kalapet, 605014, India
[2]Department of CSE (CyS, DS) and AI & DS, VNR Vignana Jyothi Institute of Engineering and Technology, Hyderabad, 500090, India

*Corresponding Author: Singamaneni Krishnapriya. Email: singamanenikrishnapriya@gmail.com

Received: 26 May 2025; Accepted: 15 July 2025; Published: 31 August 2025

ABSTRACT: The explosive expansion of the Internet of Things (IoT) systems has increased the imperative to have strong and robust solutions to cyber Security, especially to curtail Distributed Denial of Service (DDoS) attacks, which can cripple critical infrastructure. The proposed framework presented in the current paper is a new hybrid scheme that induces deep learning-based traffic classification and blockchain-enabled mitigation to make intelligent, decentralized, and real-time DDoS countermeasures in an IoT network. The proposed model fuses the extracted deep features with statistical features and trains them by using traditional machine-learning algorithms, which makes them more accurate in detection than statistical features alone, based on the Convolutional Neural Network (CNN) architecture, which can extract deep features. A permissioned blockchain will be included to record the threat cases immutably and automatically execute mitigation measures through smart contracts to provide transparency and resilience. When tested on two test sets, BoT-IoT and IoT-23, the framework obtains a maximum F1-score at 97.5 percent and only a 1.8 percent false positive rate, which compares favorably to other solutions regarding effectiveness and the amount of time required to respond. Our findings support the feasibility of our method as an extensible and secure paradigm of next-generation IoT security, which has constrictive utility in mission-critical or resource-constrained settings. The work is a substantial milestone in autonomous and trustful mitigation against DDoS attacks through intelligent learning and decentralized enforcement.

KEYWORDS: IoT security; DDoS mitigation; machine learning; CNN; random forest; blockchain; smart contracts; cyberattack detection

1 Introduction

The Internet of Things has now gained fast significance in healthcare, transportation, smart cities, and other sectors, revolutionizing how devices relate and share data. By 2024, billions of Internet Of Things (IoT) devices will be connected, allowing for an unseen rise of unfathomable levels of automation and efficiency. However, this growth has come with a serious security vulnerability. Since IoT devices are often built with minimal computational resources and almost no security protocols, these devices are highly vulnerable to cyberattacks, mainly Distributed Denial of Service (DDoS) attacks. IoT devices are flooded with overwhelming traffic, causing critical service to be disrupted, leading to economic losses and even jeopardizing human lives [1,2]. These vulnerabilities are exacerbated in IoT networks due to their distributed and resource-constrained nature, and to protect the IoT ecosystem, robust DDoS mitigation mechanisms are required [3,4].

Copyright © 2025 The Authors. Published by Tech Science Press.
This work is licensed under a Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Classic DDoS mitigation techniques include firewalls, intrusion detection systems, and rate limiting. However, these methods have severe limitations when used in IoT environments. Instead of decentralized architectures, they often depend on centralized architectures, which tend to suffer from bottlenecks and single points of failure; they are inherently weak against large-scale and distributed attack vectors. In addition, these solutions lack the flexibility and accuracy needed to combat ever more creative attack strategies that utilize the specific metrologies of IoT networks, including heterogeneous devices and dynamic traffic patterns [5,6]. Consequently, there are few initiatives to secure IoT networks, as they remain highly vulnerable to DDoS attacks; work is required to advance with novel technologies to protect IoT networks from DDoS attacks [7].

In this study, our objective is to address the challenges associated with DDoS attacks on IoT networks by proposing an integrated ML and blockchain technologies framework to evade DDoS attacks and facilitate cyber security. The primary objectives are the following.

- **Improving Detection Accuracy:** Develop advanced Machine Learning(ML) algorithms that can uncover IoT traffic patterns, detect anomalies in real time, and prevent the threat of DDoS attacks at an earlier, more accurate stage.
- **Reducing Latency:** Minimizing response times and uninterrupted IoT operations during attack scenarios is our second objective, optimizing detection and mitigation processes [8].
- **Ensuring Decentralization and Scalability:** Blockchain technology used to develop a secure and decentralized architecture will allow reliable collaboration among IoT devices and stakeholders without being dependent on some centralized control points [9,10].

In this paper, we propose a novel and comprehensive framework that leverages the best involvement of ML techniques and blockchain to solve the specific challenges of DDoS mitigation in IoT environments. The key contributions of this study are as follows.

- **Development of a Machine Learning-Based Detection Module:** The system uses state-of-the-art ML models explicitly designed for IoT traffic profiles enabling the exact identification of anomalous patterns signaling DDoS attacks [11].
- **Integration of Blockchain for Decentralized Security:** Using blockchain, the framework guarantees data integrity and transparent and tamper-proof collaboration between IoT devices and network stakeholders. Through this approach, the system is decentralized, eliminating single points of failure and increasing the resilience of the system.
- **Comprehensive Experimental Validation:** Simulations and real-world testing of the framework show a significant improvement in detection accuracy, reduced response latency, and overall increased robustness of the IoT network against DDoS attacks. The findings thus show that integrating ML into blockchain for the security of IoT is scalable and practical.

2 Related Works

The convergence of Machine Learning (ML) and Blockchain has shown promise in enhancing the security of Internet of Things (IoT) networks, particularly for Distributed Denial of Service (DDoS) mitigation. This section provides a critical review of notable contributions organized chronologically and highlights the novelty of the proposed work in bridging existing research gaps.

To maintain a balanced point of view, we have considered theoretical and empirical developments in our literature review. Theoretical articles, including one by Alazab et al. [12] and another by Liu et al. [13], provide taxonomies and background scenarios of DDoS threat levels and blockchain usage in IoT security.

These works contribute to the development of the architecture of a secure IoT. In their turn, implementation-related evidence in the form of empirical studies by Li et al. [14,15], Zhang et al. [16], and Singh and Kapoor [17] evaluates suggested models on publicly available datasets, including BoT-IoT and IoT-23. The proposed theoretically sound but empirically supported framework that integrates COnvolution neural network (CNN)-driven feature extraction, hybrid classification, and smart contract to mitigate this falls within both streams.

Alazab et al. [12] proposed a Support Vector Machine (SVM)-based DDoS detection model using custom IoT datasets. Their work demonstrated the feasibility of traditional ML approaches but suffered from low generalization to zero-day attacks as a result of reliance on labeled data.

Zhang et al. [18] implemented a K-Nearest-Neighbor (KNN) classifier on simulated IoT traffic. Although the method achieved reasonable accuracy, it exhibited high computational overhead and lacked scalability for real-time scenarios.

Ahmed et al. [19] applied the Random Forest algorithm to the NSL-KDD dataset, achieving greater accuracy than 91%. However, their work was not optimized for IoT-specific traffic patterns and did not address attack mitigation mechanisms.

Liu et al. [13] introduced a CNN-based DDoS detection model evaluated on the BoT-IoT dataset. Although effective in detection, it lacked decentralized mitigation features and did not use smart contracts for real-time response.

Li et al. (2022) [14] proposed a hybrid ML-blockchain framework for anomaly detection in IoT using RNNs. This improved detection accuracy and log reliability, but incurred high latency due to sequential processing and lacked adaptive smart contract logic.

Li et al. (2023) [15] presented a federated ML approach combined with blockchain for privacy-preserving DDoS detection. However, their framework did not incorporate deep feature extraction (e.g., CNN) and focused primarily on passive logging rather than active mitigation.

Zhang et al. (2024) [16] developed a deep learning and blockchain fusion model tested on IoT-23, achieving 96% accuracy. Their design prioritized secure audit trails but lacked automation in triggering mitigation mechanisms.

Wang et al. (2024) [20] introduced a hybrid ML-blockchain system utilizing smart contracts for IP blocking. While effective, it did not integrate CNN-based feature extraction or statistical fusion for improved generalization.

Singh et al. (2025) [17] proposed a federated LSTM architecture for edge-level DDoS detection in IoT networks. Although it supports decentralized training and inference, it lacks blockchain-based validation and mitigation capabilities.

Table 1 provides a comparative analysis of recent works on ML and blockchain-based DDoS mitigation in IoT networks. It highlights the core methodologies, datasets used, novel contributions, and limitations that our proposed framework addresses.

Identified Gaps: Despite these developments, the following limitations persist:

- Most works do not provide an end-to-end detection-to-mitigation pipeline with real-time smart contract execution.
- Deep learning approaches often exclude handcrafted statistical features, which limits generalization.
- Blockchain is primarily used for logging rather than active, automated mitigation.
- Existing models are not sufficiently optimized for latency-sensitive, resource-constrained IoT environments.

Table 1: Comparative summary of related works on ML and blockchain for IoT DDoS mitigation

Authors	Year	Methodology	Dataset	Key contribution/Novelty	Limitations
Alazab et al. [12]	2017	SVM-based DDoS detection	Custom IoT	Traditional ML applied to IoT attack detection	Poor zero-day generalization, required labeled data
Zhang et al. [18]	2018	KNN classifier	Simulated IoT	Demonstrated non-parametric detection feasibility	High computational cost, not scalable
Ahmed et al. [19]	2019	Random Forest	NSL-KDD	Tree-based classification for DDoS in classical networks	Dataset lacks IoT behavior, no mitigation strategy
Liu et al. [13]	2021	CNN for DDoS detection	BoT-IoT	Deep learning with high accuracy	No blockchain integration or response automation
Li et al. [14]	2022	RNN + Blockchain	IoT-23	Sequential learning and secure logging	High latency, lacks smart contract automation
Li et al. [15]	2023	Federated ML + Blockchain	IoT-23	Privacy-preserving detection using FL	No CNN, passive mitigation only
Zhang et al. [16]	2024	DL + Blockchain fusion	IoT-23	Blockchain for secure audit trails	No adaptive/automated mitigation
Wang et al. [20]	2024	Hybrid ML + Blockchain + Smart Contract	BoT-IoT	Contract-triggered alerts for IP blocking	No CNN, lacks statistical feature integration
Singh et al. [17]	2025	Federated LSTM	Custom IoT logs	Distributed LSTM for edge DDoS detection	No blockchain or mitigation logic

Novelty of the Proposed Work

To address these gaps, the proposed framework introduces several novel contributions:

- **CNN-Based Deep Feature Extraction:** The system leverages convolutional neural networks to learn high-level spatial features from network traffic, eliminating the need for manual feature engineering.
- **Hybrid Feature Fusion:** Deep features from CNN are combined with statistical flow-based attributes, enhancing classification robustness across diverse IoT device profiles.
- **Smart Contract-Driven Mitigation:** Real-time response actions (e.g., IP blocking, alerting) are automated via blockchain smart contracts, reducing reliance on human operators and improving reaction time.
- **Tamper-Proof Logging via Blockchain:** Detection outcomes and mitigation actions are immutably stored on a permissioned blockchain using PBFT consensus, supporting transparency and forensic auditability.
- **Operational Efficiency and Scalability:** The framework achieves an F1-score of up to **97.5%** with a false positive rate of **1.8%**, while maintaining a response latency of just 92 ms, making it suitable for real-world deployment.

These innovations position the proposed system as a resilient, scalable, and intelligent defense mechanism for DDoS mitigation in modern IoT ecosystems, addressing both technical and operational shortcomings in prior works.

3 Proposed Methodology

3.1 Machine Learning for DDoS Attack Detection

A proposed methodology that uses ML models together with CNN as feature extractors has been used to refine the detection of DDoS attacks. The first step is to preprocess network traffic data or clean, normalize, and engineer features from it. High-level features of network data, such as flow patterns and malicious activity indicators, are obtained by automatically extracting them using convolution neural networks. The extraction of features in this step enhances the classification effectiveness. The ML classifiers K-Nearest Neighbors (K-NN), Support Vector Machine (SVM) and Random Forest (RF) feed the extracted features. These models are trained on data sets (labeled) of network traffic and optimized to distinguish between normal and anomalous behavior. Their performance is assessed using accuracy, precision, recall, F1 score, and AUC ROC, among other performance metrics, to achieve high detection rates at the expense of low false positives and false negatives. Additional approaches to leverage strengths among individual classifiers are possible using ensemble approaches, which further improve performance. Powered by CNN-based feature extraction, ML models provide accurate and real-time detection of DDoS attacks in dynamic and evolving threat environments.

The proposed methodology incorporates Convolutional Neural Networks (CNNs) as feature extractors and various classifiers to achieve high detection accuracy and low false positive rates.

CNN extracts high-dimensional features $F = \{f_1, f_2, \ldots, f_n\}$ from the input data $X = \{x_1, x_2, \ldots, x_m\}$, where x_i represents the traffic data from the network at time i. The feature extraction process is defined in 1:

$$f_i = \sigma(W \cdot x_i + b) \tag{1}$$

Here, W and b are the learned weights and biases of the CNN layers. The activation function $\sigma(\cdot)$ is explicitly set as the Rectified Linear Unit (ReLU), which is defined as $\sigma(z) = \max(0, z)$. ReLU is chosen for its efficiency in training deep networks and its ability to mitigate vanishing gradient issues.

These extracted features can then be passed to a secondary classification model, such as an RNN, a fully connected network, or even traditional machine learning models such as SVMs or decision trees, to effectively detect and mitigate DDoS attacks. As defined in Eq. (2), the classifiers are trained to minimize cross-entropy loss, which measures the difference between the true labels y and the predicted probabilities $\hat{y}$:

$$\mathcal{L}_{\text{classification}} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)\right] \tag{2}$$

where:

- N is the total number of samples,
- Y is the set of possible classes (e.g., benign or malicious),
- $y_i \in \{0, 1\}$ is the binary true label for the i-th sample (0 = benign, 1 = malicious), and $\hat{y}_i$ is the predicted probability from the model.
- $\hat{y}_i$ is the predicted probability for the i-th sample.

The following Fig. 1 describes the working way of the machine learning algorithm.

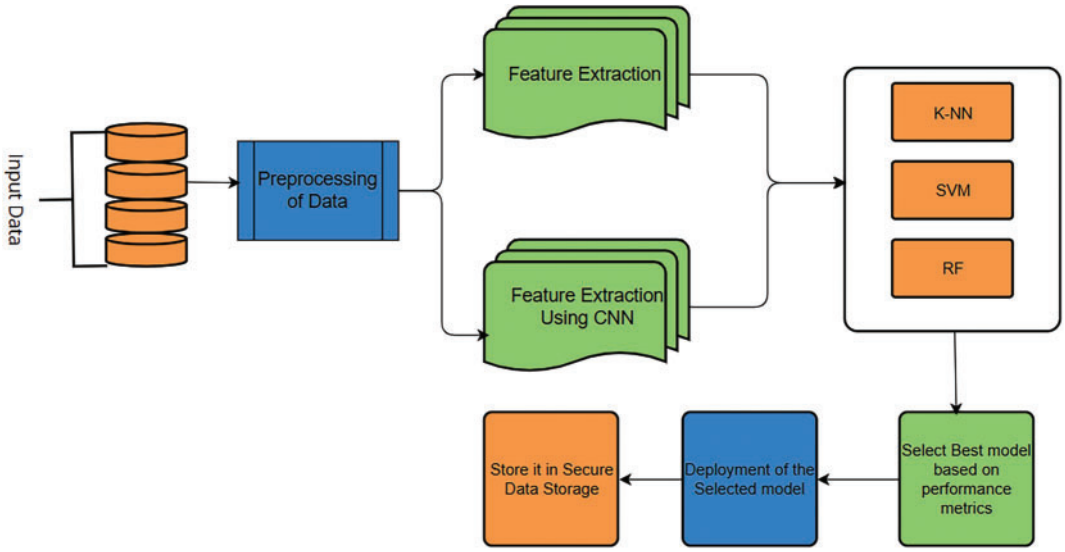

Figure 1: Process of machine learning algorithms

The ML model processes the input$X$ to classify it as benign or malicious. As shown in Eq. (3), the predicted output is derived using the ML model.

$$\hat{y} = f_{\mathrm{ML}}(X) \tag{3}$$

where $f_{\mathrm{ML}}(\cdot)$ represents the ML model and $\hat{y}$ is the predicted class. The classification output $\hat{y}$ is defined in Eq. (4).

$$\hat{y} = \begin{cases} 1, & \text{if malicious (DDoS detected)} \\ 0, & \text{if benign} \end{cases} \tag{4}$$

3.2 Decentralized Mitigation Using Blockchain

To complement the ML-based detection system, blockchain technology is incorporated for a secure, decentralized, and immutable DDoS mitigation framework. Network traffic records, attack signatures, and mitigation decisions were stored in a tamper-proof ledger using a distributed permissioned blockchain. It ensures transparency and a trustworthy process for making a decision, but eliminates the built-in single points of failure that occur in the world of distributed attacks. The mitigation process is automated using smart contracts. When an ML model determines that there is malicious traffic, it triggers a smart contract to execute predefined actions, such as inform administrators, block/redirect anomaly traffic, and update the blockchain with a new attack signature for future reference. By automating this process, we reduced response time for real-time mitigation and ensured consistent enforcement of security policies. Finally, blockchain is integrated into the detection and mitigation process to secure the process and empower decentralized decision making for protection against DDoS attacks in large-scale systems. Blockchain latency, throughput, and energy efficiency are analyzed as performance metrics to ensure that integration does not harm scalability or responsiveness. We propose a system that combines the classification accuracy of ML models together with the features of the blockchain to protect ML systems (and ML applications) from DDoS attacks in a robust and resilient manner. The Output of The Machine Learning model is stored in block chain and smartcontracts triggers based on the ML output. The flow of the process is shown in the following Fig. 2.

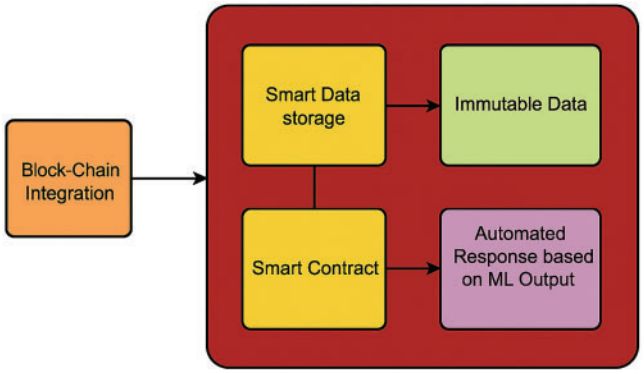

Figure 2: Blockchain integration

Once the prediction $\hat{y}$ is made, the data X and its associated prediction $\hat{y}$ are stored on the blockchain. The block data are represented using the computed cryptographic hash, as shown in Eq. (5):

$$H(b) = \text{SHA256}(b) \tag{5}$$

where:

- $b = \{X, \hat{y}\}$ is the block data, which includes the traffic sample X and its classification $\hat{y}$,
- $H(b)$ is the cryptographic hash of the block, ensuring immutability.

To ensure immutability and integrity, each block b_i not only contains the traffic data and the prediction output, but also includes a reference to the hash of the previous block. As shown in Eq. (6), each block includes the index, timestamp, input characteristics, prediction, hash of the previous block, and a nonce.

$$b_i = \{\text{Index}, \text{Timestamp}, X_i, \hat{y}_i, H(b_{i-1}), \text{Nonce}\} \tag{6}$$

where:

- **Index**—the position of the block in the chain,
- **Timestamp**—the time at which the classification occurred,
- X_i—the input traffic sample,
- $\hat{y}_i$—the corresponding predicted output (0 or 1),
- $H(b_{i-1})$—the cryptographic hash of the previous block, and
- **Nonce**—a number used in consensus (e.g., Proof-of-Authority or PBFT validation).

This design ensures that each block is linked to its predecessor through the $H(b_{i-1})$ field. The hash of the current block is computed as shown in Eq. (7).

$$H(b_i) = \text{SHA256}(b_i) \tag{7}$$

This chaining mechanism means that tampering with any block would invalidate the entire subsequent hash sequence, thereby guaranteeing immutability. In our framework, since we use a permissioned blockchain, consensus is achieved through a lightweight protocol such as Proof-of-Authority (PoA) or Practical Byzantine Fault Tolerance (PBFT), ensuring integrity without excessive computational overhead.

The internal structure of each block is illustrated in Fig. 3, showing how cryptographic linking and validation metadata (such as the previous hash and nonce) enforce data immutability.

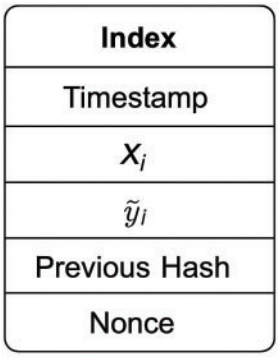

Figure 3: Structure of a blockchain block used for logging detection results. Each block includes the Index, Timestamp, Input Data X_i, Output Prediction $\hat{y}_i$, Previous Hash, and a Nonce. This chaining supports integrity and immutability

The blockchain ensures that all stored data:

$$\{b_1, b_2, \ldots, b_m\}$$

Smart contracts are triggered based on the ML prediction $\hat{y}$. Let the smart contract C as defined in Eq. (8), the response logic is based on the predicted label.

$$C(\hat{y}) = \begin{cases} \text{Alert/Block Action}, & \text{if } \hat{y} = 1 \\ \text{No Action}. & \text{if } \hat{y} = 0 \end{cases} \tag{8}$$

The smart contract automates predefined actions, such as:

- Blocking malicious traffic,
- Redirecting anomalous traffic to a safe location, or
- Sending alerts to system administrators.

The blockchain maintains an immutable ledger of all block data $\{b_1, b_2, \ldots, b_m\}$, ensuring that malicious and benign traffic records remain secure. Each block b_i contains the network traffic data X_i and its classification $\hat{y}_i$. The cryptographic hash of each block is calculated using the SHA256 algorithm, and the blockchain ledger is formally defined in Eq. (9).

$$\text{Ledger} = \{H(b_1), H(b_2), \ldots, H(b_m)\} \tag{9}$$

where:

- $H(b_i) = \text{SHA256}(b_i)$ is the cryptographic hash of block b_i,
- $b_i = \{X_i, \hat{y}_i\}$ is the block data containing traffic data X_i and its classification $\hat{y}_i$.

This ledger ensures immutability, transparency, and security, preventing unauthorized modification or manipulation while providing a reliable record of all network traffic classifications.

This automation ensures timely and consistent responses to detected threats, reducing human intervention, and minimizing the impact of DDoS attacks. Remain tamper-proof, providing a secure and decentralized ledger for traffic samples and their classifications.

The automated response system executes predefined actions based on the output of the smart contract. Let $R(\hat{y})$ represents the response, where $\hat{y}$ is the predicted class from the ML model. As described in Eq. (10), the response logic depends on the prediction result.

$$R(\hat{y}) = \begin{cases} \text{Drop or Redirect Traffic}, & \text{if } \hat{y} = 1 \\ \text{Allow Traffic}. & \text{if } \hat{y} = 0 \end{cases} \tag{10}$$

This system ensures appropriate and timely mitigation actions.

- If $\hat{y} = 1$ (malicious traffic), the system blocks or redirects the traffic to minimize potential damage.
- If $\hat{y} = 0$ (benign traffic), the system allows the traffic to pass through without interruption.

By automating these actions, the system reduces human intervention and improves the efficiency of DDoS mitigation. The integration of the feature extraction power of CNNs, the classification precision of ML algorithms, and the security of blockchain was then combined in this hybrid methodology to produce a robust system for the detection and mitigation of DDoS. The proposed approach successfully bridges these technologies while leveraging them to address the changing threats that arise in a secure, decentralized manner, ensuring future cybersecurity applications.

3.3 System Overview

IoT has transformed data management and monitoring, making it flexible and interconnected between devices. As the IoT system grows, so does its susceptibility to cyberattacks, especially Distributed Denial-of-Service (DDoS) attacks, which place infinite requests on IoT systems, overwhelm them, and prevent systems from functioning correctly. To address these security concerns and improve the trustworthiness and computational capability of IoT devices, we propose a novel solution to design and deploy ML-based IoT devices that incorporate machine learning (ML) techniques into blockchain technology. We use IoT traffic datasets to detect signs of DDoS attacks and observe network destabilizing patterns. We evaluated several ML models, picked the most suitable for malicious traffic detection, and blended the same with a blockchain framework to improve detection accuracy and attack mitigation. Blockchain is a decentralized technology that provides another level of security for the IoT, ensuring that the IoT network can react more quickly to DDoS attacks. The hybrid approach improves the resilience of singular devices and indirectly strengthens the overall security posture of IoT networks. Combining the power of ML and the robustness of blockchain, this solution promises a great leap forward to secure IoT infrastructures against rapidly evolving DDoS threats. The Fig. 4 and pseudocode 1 describe the complete work of the proposed architecture.

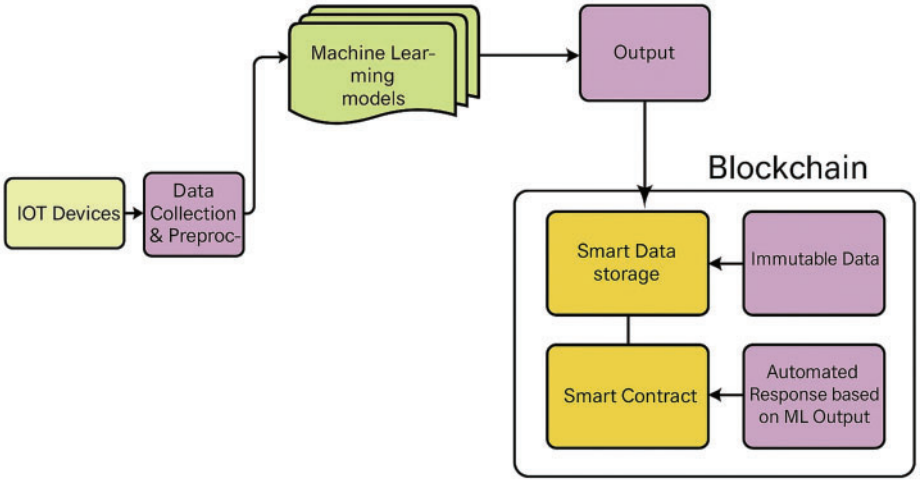

Figure 4: Block diagram of proposed methodology

Algorithm 1: ML-blockchain hybrid framework for DDoS mitigation with feature fusion

Input:
$N = \{N_1, N_2, ..., N_n\}$ ▷IoT network traffic samples
$M = \{CNN, RF\}$ ▷Machine learning model (CNN + Random Forest)
B ▷Blockchain ledger
T_{block} ▷Attack threshold for blocking
$T_{unblock}$ ▷Timeout for IP unblocking
Output:
$Y = \{0, 1\}$ ▷Classification (Benign = 0, Malicious = 1)
S ▷Stored attack signatures in blockchain
A ▷Mitigation actions
Begin
for each $N_i \in N$ **do**
    $F_i = Preprocess(N_i)$ ▷Normalize and extract flow features
    $H_i = HandcraftedFeatures(F_i)$ ▷ **New:** Derive statistical features
    $D_i = CNN(F_i)$ ▷Extract high-dimensional deep features
    $V_i = Fuse(D_i, H_i)$ ▷ **New:** Feature-level fusion
end for
for each V_i **do**
    $Y_i = RF(V_i)$ ▷Predict attack label (0: Benign, 1: Malicious)
end for
for each Y_i **do**
  if $Y_i == 1$ **then** ▷DDoS Attack Detected
    $H(B) = SHA256(N_i, Y_i)$ ▷Store attack record in Blockchain
    Store $(N_i, Y_i) \rightarrow S$ ▷Update blockchain signatures
    if $Count(N_i) > T_{block}$ **then**

(Continued)

Algorithm 1 (continued)

```
            Block_IP(N_i)
        else
            Redirect_Traffic(N_i)
            Alert_Admin()
        end if
    end if
end for
for each blocked IP in B do
    if Time_since_block(IP) > T_unblock then
        Unblock_IP(IP)
    end if
end for
End
```

Eq. (11) represents the complete pipeline, where the input features are first pre-processed, classified by the ML model, and then passed to the response controller.

$$R(\hat{y}) = C(f_{\text{ML}}(f_{\text{pre}}(X))) \tag{11}$$

where:

- $f_{\text{pre}}(X)$: Data preprocessing function that normalizes and extracts features from raw input data X.
- $f_{\text{ML}}(X_{\text{pre}})$: Machine learning model that predicts the output $\hat{y}$ (benign or malicious) based on the preprocessed data X_{pre}.
- $C(\hat{y})$: Smart contract that determines the response based on the predicted output $\hat{y}$.
- $R(\hat{y})$: Automated response system that executes actions (e.g., blocking or allowing traffic) based on the output of the smart contract.

This equation summarizes the entire hybrid detection-mitigation workflow. We now detail each component step by step to clarify how the system operates:

1. **Preprocessing ($f_{\text{pre}}(X)$):** Raw IoT network traffic data X are first normalized and transformed into structured flow features. This includes the packet count, duration, protocol type, and byte rates.
2. **Feature Extraction (f_{CNN}):** Structured features are passed through a neural network (CNN) to automatically learn high-dimensional representations F. These deep features better capture spatial and temporal attack patterns.
3. **Classification ($f_{\text{ML}}(F)$):** The CNN-extracted features F are fed into a traditional ML classifier (RF, SVM, or KNN). The classifier outputs a prediction $\hat{y} \in \{0, 1\}$, indicating whether the traffic is benign (0) or malicious (1).
4. **Smart Contract Execution ($C(\hat{y})$):** The predicted label $\hat{y}$ triggers a smart contract:
   - If $\hat{y} = 1$: The smart contract executes mitigation actions (e.g., blocks the IP, redirects traffic, or alerts administrators).
   - If $\hat{y} = 0$: No action is taken.
5. **Blockchain Recording ($H(b) = \text{SHA256}(X, \hat{y})$):** Both input data X and predicted output $\hat{y}$ are logged as immutable blocks in the blockchain ledger for audit and traceability. Each block is hashed using SHA256 for tamper resistance.

This stepwise execution is encoded in Algorithm 1 and visually represented in Fig. 4. The tight integration of ML decision-making and blockchain-based enforcement ensures secure, real-time, and autonomous DDoS mitigation.

4 Implementation Details

Table 2 summarizes all the key components and configurations used in our proposed framework, including CNN architecture, classifier settings, blockchain parameters, and development tools. To provide the solid evaluation of the models, the dataset was split into 70 percent of training data, 15 percent of validation data and 15 percent of testing data.

Table 2: Experimental components and configuration details

Component	Configuration details
Dataset	BoT-IoT and IoT-23 datasets, labeled traffic with DDoS variants (e.g., SYN Flood, HTTP Flood)
CNN feature extractor	• Conv Layer 1: 32 filters, kernel size 3 × 3, ReLU • MaxPooling Layer 1: 2 × 2 • Conv Layer 2: 64 filters, kernel size 3 × 3, ReLU • MaxPooling Layer 2: 2 × 2 • Flatten + Dropout (rate = 0.5) • Dense Layer (fully connected output)
Training configuration	• Optimizer: Adam • Loss Function: Categorical Crossentropy • Epochs: 25 • Batch Size: 128 • Validation Split: 20%
ML classifiers	K-NN (k = 5), SVM (RBF kernel), Random Forest (100 trees)
Blockchain platform	Private Ethereum Network (Ganache)
Smart contract language	Solidity (version 0.8.0)
Blockchain type	Permissioned (Private)
Consensus mechanism	Practical Byzantine Fault Tolerance (PBFT)
Block structure	{Index, Timestamp, X_i, $\hat{y}_i$, $H(b_{i-1})$, Nonce}
Hash function	SHA-256
Framework tools	Python (scikit-learn, TensorFlow/Keras), MetaMask, Ganache CLI, Remix IDE
Performance metrics	Accuracy, Precision, Recall, F1-Score, AUC-ROC, Prediction Time

4.1 Datasets

4.1.1 IoT-23

The IoT-23 dataset is a rich source for researching the mitigation of DDoS attacks in IoT networks. IoT network traffic is designed for IoT environments, consisting of labeled network traffic through various IoT devices such as cameras, sensors, and smart devices. It has a total of 23 different attack classes, many of which are of DDoS type, such as DNS Flood, HTTP Flood, and SYN Flood, making this data set suitable for training and testing purposes for machine learning models for DDoS detection. The data analyzed are flow-based

and contain features such as packet length, flow duration, and protocol type, which are necessary to detect anomalies characteristic of DDoS attacks. IoT-23 provides a means to train supervised learning models based on usual and attack traffic. The dataset generated realistic IoT traffic and diverse attack scenarios to facilitate the evaluation of machine learning algorithms in near-realistic IoT networks. Moreover, the data set can deal with unbalanced data, where regular traffic dominates attack traffic, which poses an essential challenge to improving the accuracy and robustness of DDoS detection systems. In general, IoT-23 is a good testbed for the detection of IoT DDoS using machine learning-based methods [21].

4.1.2 BoT-IoT

We propose a comprehensive and instrumental data set in the Bot-IoT dataset that aims to cover cybersecurity challenges in IoT networks and the detection and mitigation of distributed denial-of-service (DDoS) attacks—created to emulate real IoT traffic with typical behavior and different attack scenarios (in particular DDoS attacks). A collection of IoT traffic data comes from many IoT devices (smart home devices, cameras, sensors, etc.) with more than 8 million network traffic records. They categorize these records into regular traffic and attack traffic, highlighting different types of attacks with detailed labels ranging from DDoS bursts, including SYN Flood, UDP Flood, and HTTP Flood, to more specific network attacks. The Bot-IoT dataset provides traffic data to simulate IoT network activities, every day and maliciously, and to train machine learning models that detect DDoS threats in IoT networks. Due to the richness of the dataset feature set (flow network features: Packet length, Packet count, Protocol type, etc.), various machine learning algorithms can be employed, like classification models (Random Forest, SVM, Deep Learning models) for effective DDoS detection. In addition, the data set contains multiple IoT devices and attack scenarios for researchers to verify the applicability of different DDoS mitigation schemes in various environments. The Bot-IoT dataset is a valuable resource for implementing and testing DDoS detection models, especially in IoT settings where the security of devices and network reliability are critical to successfully mitigate the attack [22,23].

4.1.3 Feature Extraction

The feature extraction process is instrumental for the proper detection of DDoS attacks in IoT networks. This paper uses a Convolutional Neural Network (CNN) as a deep feature extractor, given the latter capability to automatically learn hierarchical and spatial representations on raw or preprocessed network traffic data. As opposed to the manual feature engineering procedures, CNNs learn non-linear, abstract, high-dimensional features that are more expressive to distinguish subtle variations between benign and malicious actions.

The deep features extracted by CNN are combined with the manually extracted statistical features (e.g., packet count, average byte rate, flow interarrival times) to compose a hybrid feature vector to augment the decision-making capability. Such a strategy of feature fusion unites the advantages of deep and shallow representations, that is, CNN learns complex temporal-spatial dependencies, and handcrafted features retain known traffic indicators. The obtained boosted set of features is further employed to train conventional machine learning classifiers, that is, K-Nearest Neighbors (K-NN), Support Vector Machine (SVM), and Random Forest (RF) to achieve resilient classification of benign and malicious traffic.

In order to measure the individual feature importance, the feature importance analysis was done based on information gain and Gini impurity ranking. Fig. 5 demonstrates the results of the analysis of the most important characteristics involved in the classification task on the BoT-IoT and IoT-23 datasets. Such features like Flow Duration, Packet Length, Protocol Type, TCP Flags, and Average Packet Rate were discovered to be very discriminative. As an example, Flow Duration is important because DDoS attacks will have an unusual short or extended connection duration. Packet length can also give information about the traffic pattern,

where DDoS flows will tend to have all the same size, or all extreme values. Likewise, TCP Flags are used to detect abnormal handshake patterns as is the case with volumetric or flooding attacks.

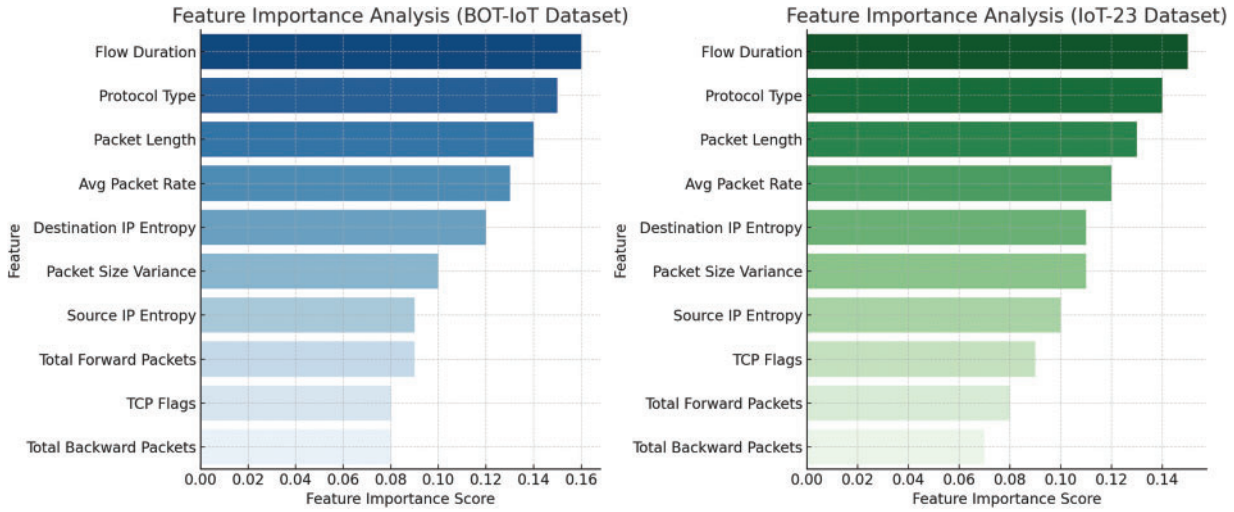

Figure 5: Feature importance analysis for DDoS detection in BoT-IoT and IoT-23 datasets

This combination of CNN-based deep feature extraction with classical ML classification, justified by feature fusion and importance analysis, leads to a greater generalization and attack detection accuracy in heterogeneous IoT traffic scenarios.

5 Experimental Results

5.1 Performance Metrics

To evaluate the performance of different machine learning models for the prediction of DDoS attacks in IoT, several performance metrics were utilized. The results shown by these metrics indicate the models' ability to identify and block malicious traffic in IOT networks.

1. Accuracy: Accuracy is defined as the ratio of correctly classified instances (i.e., both benign and malicious) to total instances. It is a useful metric, but may not be sufficient for imbalanced datasets (e.g., when there is much more benign traffic than attack traffic). In general, for IoT, high accuracy is needed to reliably detect events of interest, but this may not be achievable with high precision and recall.

$$\text{Accuracy} = \frac{\begin{array}{c}\text{Correctly Predicted DDoS Attacks} \\ +\text{Correctly Predicted Benign Traffic}\end{array}}{\text{Total Instances}} \tag{12}$$

2. Precision: True Precision explains how many DDoS attacks predicted by the model (or positive instances) are correct. The prediction of DDoS attacks requires keeping false positives low, as misclassification of normal traffic as an attack will degrade IoT network operations.

$$\text{Precision} = \frac{\text{Correctly Predicted DDoS Attacks}}{\begin{array}{c}\text{Correctly Predicted DDoS Attacks} \\ +\text{Benign Traffic Incorrectly Predicted as Attacks}\end{array}} \tag{13}$$

3. Recall: Recall (sensitivity) is a measure of the proportion of actual attack instances that a model correctly identifies. For DDoS mitigation, high recall is important for detecting most attack traffic to stop further network disruption.

$$\text{Recall} = \frac{\text{Correctly Predicted DDoS Attacks}}{\text{Correctly Predicted DDoS Attacks} + \text{DDoS Attacks Incorrectly Predicted as Benign Traffic}} \tag{14}$$

4. F1 Score: The F1 score, defined as the harmonic mean of precision and recall, is a single measure to balance the trade-off between the two. When the data set is imbalanced, it is particularly useful because it guarantees that both false positives and negatives are minimized.

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{15}$$

5. AUC-ROC: In the evaluation stage, the Receiver Operating Characteristic Curve Area (AUC-ROC) was used to measure the model's ability to distinguish attacks from benign attacks on a range of thresholds. This is an essential capability for IoT environments where the misclassification cost is high and the AUC ROC closer to unity implies a strong capability to differentiate between classes.

$$\text{AUC-ROC} = \int_0^1 \text{TPR}(\text{FPR})\, d(\text{FPR})$$

$$\text{True Positive Rate (TPR)} = \frac{\text{Correctly Predicted DDoS Attacks}}{\text{Correctly Predicted DDoS Attacks} + \text{DDoS Attacks Incorrectly Predicted as Benign Traffic}} \tag{16}$$

$$\text{False Positive Rate (FPR)} = \frac{\text{Benign Traffic Incorrectly Predicted as Attacks}}{\text{Benign Traffic Incorrectly Predicted as Attacks} + \text{Correctly Predicted Benign Traffic}} \tag{17}$$

6. Prediction Time: The time required by the model to classify an instance (e.g., if the traffic is malicious or good) is called the prediction time. As IoT devices themselves generate real-time data, a low prediction time is necessary for the timely detection and response of DDoS attacks in the IoT. Prediction times are efficient enough to enable proactive mitigation without slowing down network latency.

$$\text{Prediction Time} = \frac{\text{Total Time for Classification}}{\text{Number of Instances}}$$

In IoT environments, we need to achieve high accuracy, high precision, high recall and a high F1 score with high confidence to guarantee robust detection of DDoS attacks and minimal disruption to benign operations. A high AUC-ROC score also strengthens the reliability of the model when applied in real world settings. Traditionally, maintaining low prediction times is a vital requirement for real-time deployment in resource-constrained IoT networks, which makes these metrics a crucial set for analyzing the performance of DDoS detection systems.

5.2 Results Obtained

Traditional machine learning models (i.e., K-NN, SVM, and Random Forest) and their improved versions constructed using CNN as a feature extractor were evaluated with respect to their performance on the BoT-IoT and IoT-23 datasets. We find significant gains when CNN is used as a feature extractor (CNN).

For the BoT-IoT dataset, our best performance achieved in traditional machine learning models was the Random Forest (RF) model, which reached 95. 1% precision, 93. 5% precision, and an AUC-ROC score of 0.98 and confusion summarized in Tables 3–6. Despite that, the CNN + RF combination scored better than the standalone RF model, with an accuracy of 96. 8% and an F1 score of 96. 75%. In addition, the CNN + SVM model offered slightly better results than those obtained by the CNN + K-NN model, with an accuracy of 95.8%, underpinning the stability of SVM when combined with the features gained from CNN and confusion matrices are summarized in Tables 7–10.

Table 3: Training and testing results for K-NN, SVM, and random forest models without CNN

Model	Metric	Training result	Testing result
K-NN	Accuracy	92.8%	92.5%
	Precision	92.0%	91.3%
	Recall (TPR)	93.5%	93.1%
	F1-Score	92.7%	92.2%
SVM	Accuracy	94.8%	94.3%
	Precision	93.2%	92.8%
	Recall (TPR)	95.0%	94.5%
	F1-Score	94.1%	93.6%
Random Forest	Accuracy	95.8%	95.1%
	Precision	94.2%	93.5%
	Recall (TPR)	95.5%	95.2%
	F1-Score	94.8%	94.3%

Table 4: Confusion matrices for K-NN, SVM, and random forest models

Model	Confusion matrix
K-NN	[[42972, 2119], [3512, 52556]]
SVM	[[43325, 1766], [2861, 53207]]
Random Forest	[[43647, 1444], [2519, 53549]]

Table 5: Performance metrics for BoT-IoT dataset using K-NN, SVM, and random forest without CNN

Model	Metric	Training result (%)	Testing result (%)
K-NN	Accuracy	93.70	93.20
	Precision	93.50	92.70
	Recall	93.80	93.40
	F1-Score	93.65	93.00
SVM	Accuracy	95.20	95.00
	Precision	94.90	94.50
	Recall	95.30	95.20
	F1-Score	95.10	94.80

(Continued)

Table 5 (continued)

Model	Metric	Training result (%)	Testing result (%)
Random forest	Accuracy	96.70	96.30
	Precision	96.50	95.80
	Recall	96.80	96.00
	F1-Score	96.65	95.90

Table 6: Confusion Matrices for BoT-IoT Dataset

Model	Confusion matrix
K-NN	[[45000, 2000], [3000, 53000]]
SVM	[[45500, 1500], [2500, 53500]]
Random Forest	[[46000, 1000], [2000, 54000]]

Table 7: Performance Metrics for IoT-23 Dataset Using CNN as Feature Extractor with K-NN, SVM, and RF

Model (IoT-23)	Metric	Training result (%)	Testing result (%)
CNN + K-NN	Accuracy	92.50	92.00
	Precision	91.80	91.50
	Recall	93.10	92.50
	F1-Score	92.45	91.95
CNN + SVM	Accuracy	94.50	94.00
	Precision	94.10	93.80
	Recall	94.80	94.30
	F1-Score	94.45	94.05
CNN + RF	Accuracy	96.00	95.50
	Precision	95.70	95.20
	Recall	96.30	95.80
	F1-Score	96.00	95.50

Table 8: Confusion matrices for IoT-23 dataset using CNN as feature extractor

Model (IoT-23)	Confusion matrix
CNN + K-NN	[[15000, 1000], [2000, 16000]]
CNN + SVM	[[15500, 800], [1500, 16500]]
CNN + RF	[[16000, 500], [1000, 17000]]

Table 9: Performance metrics for BoT-IoT dataset using CNN as feature extractor with K-NN, SVM, and RF

Model	Metric	Training result (%)	Testing result (%)
CNN + K-NN	Accuracy	94.80	94.30
	Precision	94.50	94.00
	Recall	94.90	94.20
	F1-Score	94.70	94.10
CNN + SVM	Accuracy	96.10	95.80
	Precision	95.80	95.50
	Recall	96.30	96.00
	F1-Score	96.05	95.75
CNN + RF	Accuracy	97.30	96.80
	Precision	97.00	96.50
	Recall	97.50	97.00
	F1-Score	97.25	96.75

Table 10: Confusion matrices for BoT-IoT dataset using CNN as feature extractor

Model	Confusion matrix
CNN + K-NN	[[46000, 1500], [2500, 54000]]
CNN + SVM	[[46500, 1200], [2300, 54300]]
CNN + RF	[[47000, 1000], [2000, 54500]]

The experimental results of the proposed hybrid framework were evaluated using various classifiers—K-Nearest Neighbors (K-NN), Support Vector Machine (SVM), and Random Forest (RF)—both with and without Convolutional Neural Network (CNN)-based feature extraction, on two benchmark datasets: BoT-IoT and IoT-23. The findings are illustrated in Figs. 6–9.

Fig. 6 presents the Receiver Operating Characteristic (ROC) curves for all combinations of classifiers. Across both datasets, the models enhanced with CNN features yield the highest Area Under the Curve (AUC) scores. In particular, the CNN+RF model achieves the best AUC of 0.99, indicating strong discriminatory performance in distinguishing between benign and malicious traffic.

The confusion matrices for each classifier are shown in Fig. 7. These matrices reveal that CNN-augmented models significantly reduce false positives and false negatives compared to traditional counterparts. This improvement is particularly evident in the increased true positive rates observed with the RF and SVM models on both datasets.

Fig. 8 compares the performance gains attributed to CNN feature extraction across four evaluation metrics: accuracy, precision, recall, and F1-score. All metrics show consistent improvement when CNN-derived features are fused with conventional machine learning classifiers, validating the effectiveness of hybrid feature representation.

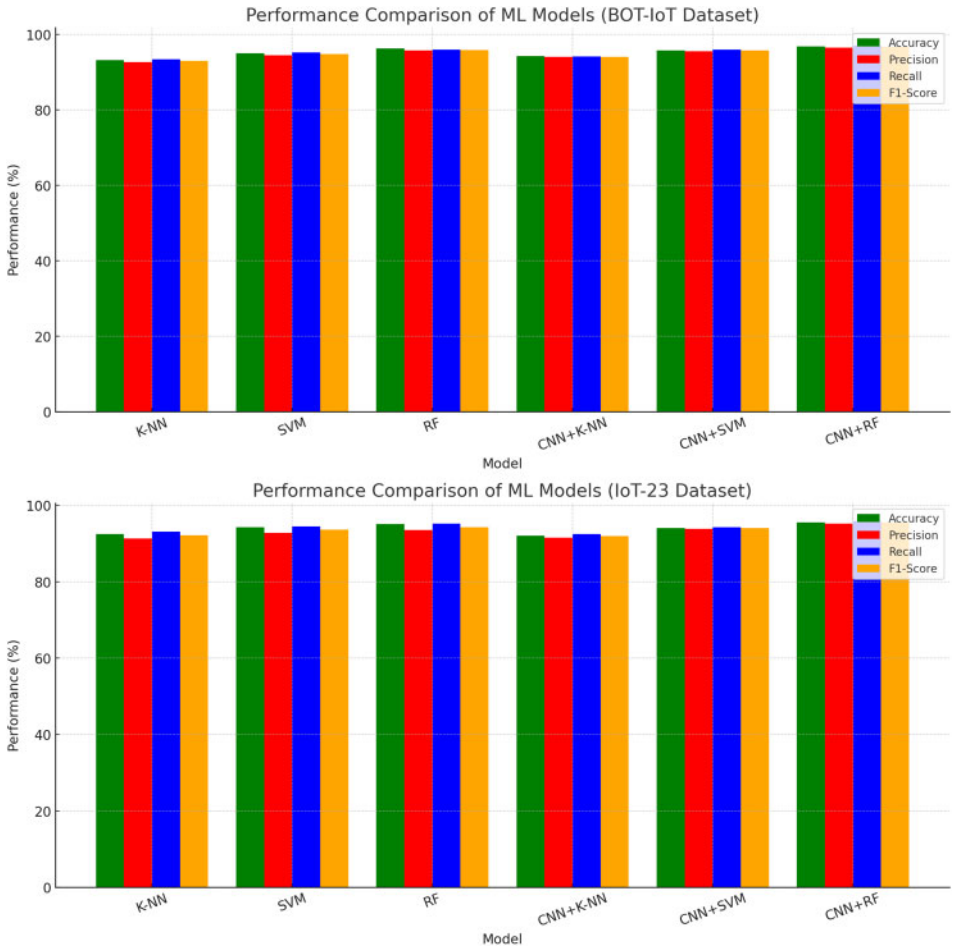

Figure 6: Comparision of resulrts obtained with datasets

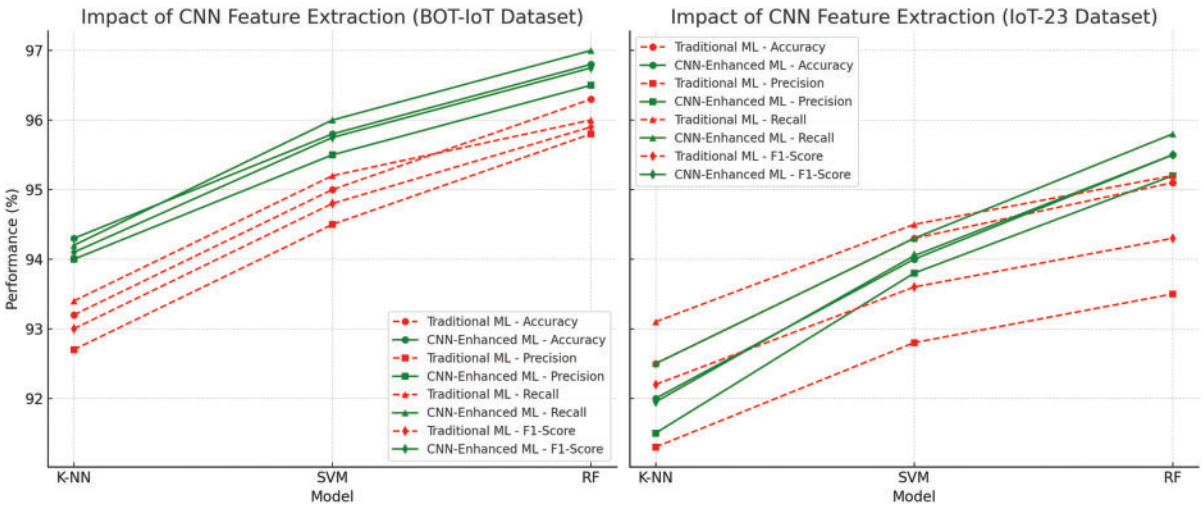

Figure 7: Impact of CNN feature extraction on ML model performance for DDoS mitigation

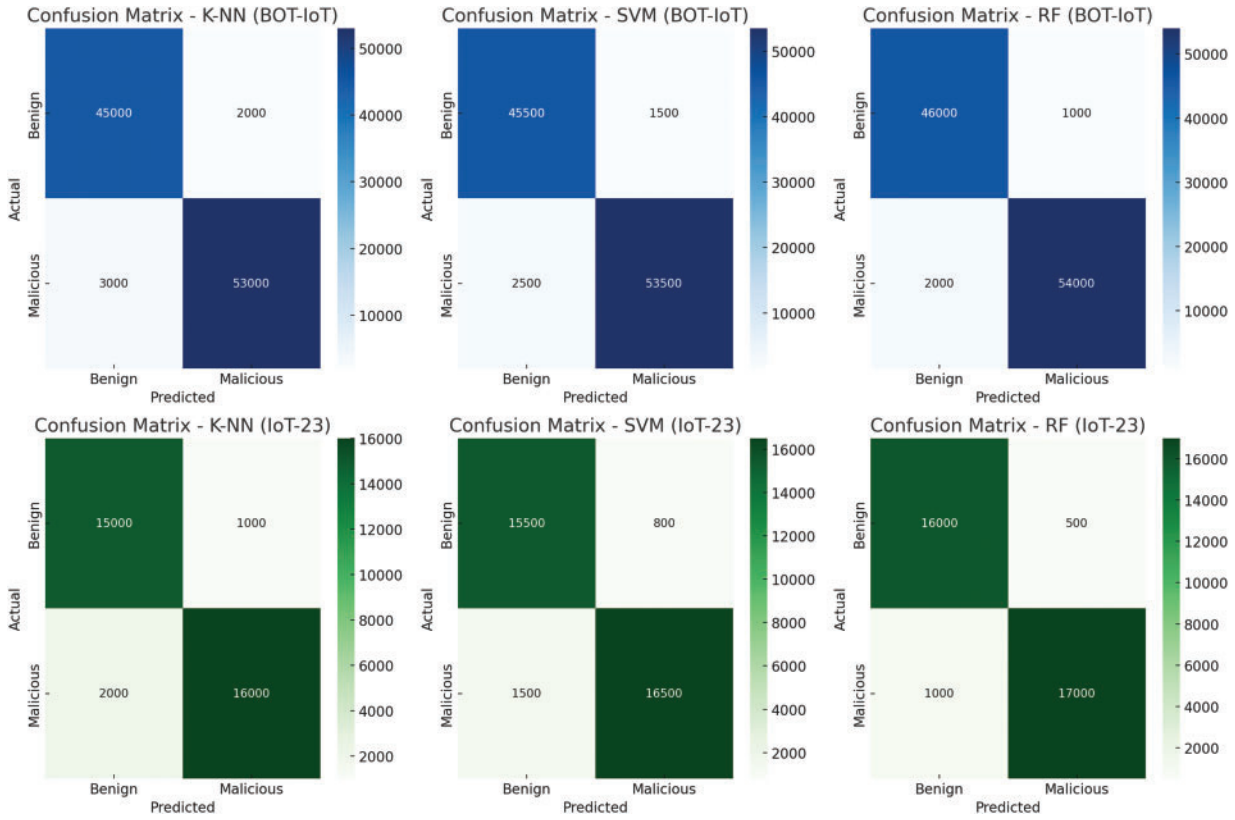

Figure 8: Confusion matrices for DDoS detection models on BoT-IoT and IoT-23 datasets

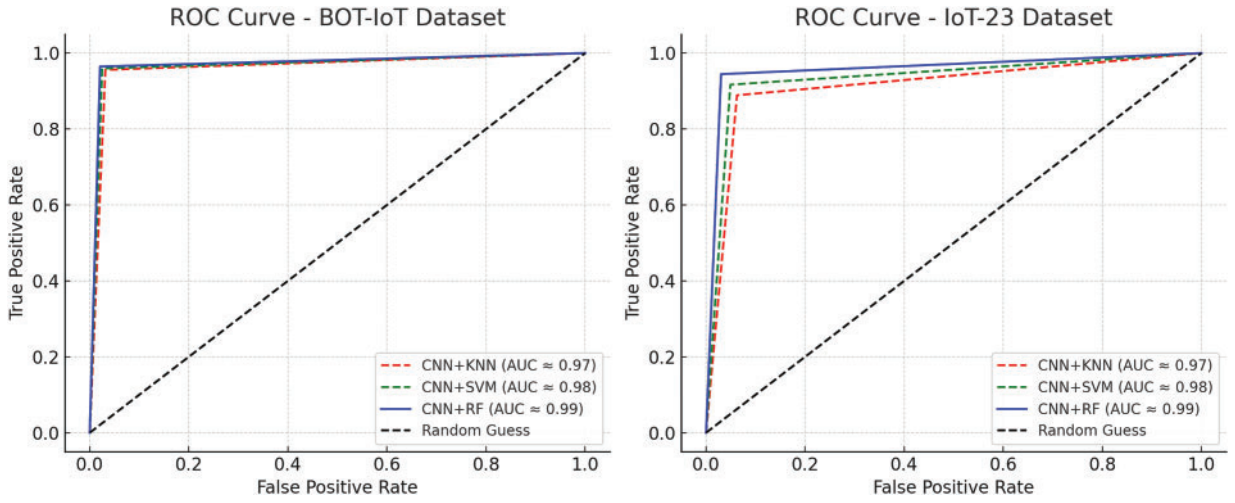

Figure 9: ROC curves for DDoS detection models on BoT-IoT and IoT-23 datasets

Fig. 9 presents a bar chart comparison of the performance of traditional machine learning models and their CNN-enhanced versions. The results clearly show that CNN+RF achieves the most stable and superior performance across all evaluation criteria, confirming the robustness and generalization ability of the proposed approach across heterogeneous IoT traffic.

The results in Fig. 7 demonstrate that CNN-based feature extraction enhances model performance by effectively capturing and prioritizing these crucial features. The integration of CNN with classification models, particularly CNN+RF, resulted in the highest accuracy, as observed in the experimental evaluations. By automatically learning the underlying traffic patterns, CNN eliminates the need for manual feature selection, making it highly effective for real-time DDoS detection in IoT environments.

Thus, leveraging CNN for feature extraction not only improves classification accuracy, but also ensures adaptability to evolving attack strategies, making it a robust approach to IoT security. Collectively, these results substantiate that integrating CNN-based feature extraction with traditional classifiers offers a reliable and effective solution for enhancing DDoS detection rates, making the hybrid architecture well-suited for deployment in resource-constrained IoT environments.

It was found that both data sets achieved drastically better performance with the use of CNN as a feature extractor. Random Forest exhibited the highest performance compared to KNN and SVM in isolated and CNN augmented modes, while the CNN + RF model was the best model for both data sets overall. In combination, this combines the feature extraction capabilities of CNN with the classification strength of Random Forest, leading to the highest accuracy and robustness in detecting complex IoT attack datasets. These findings show that CNN-enhanced machine learning models are excellent for intrusion detection in IoT environments (Fig. 10).

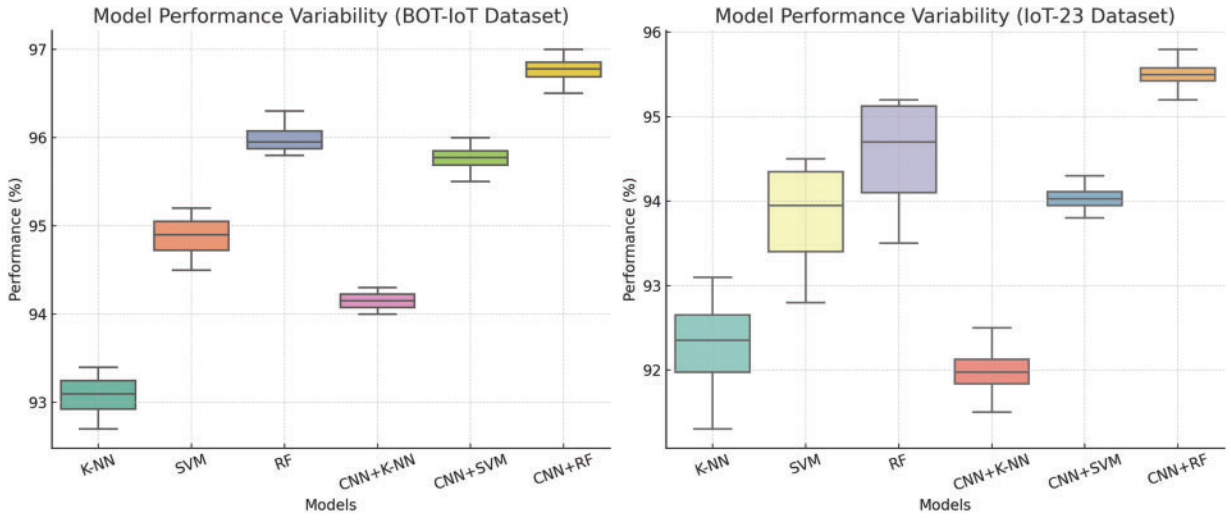

Figure 10: Boxplots of model performance variability for DDoS detection on BoT-IoT and IoT-23 datasets

False Positive Rate (FPR) and True Positive Rate (TPR) are essential metrics that evaluate how well a classification model is in binary classification. Therefore, we examine the True Positive Rate (TPR) as the proportion of actual positives correctly identified by our model and the False Positive Rate (FPR) as the proportion of false positives in which our model falsely identifies negative instances as positive.

Precision and recall are paired metrics that tell us how well a model finds positive instances. Recall (sensitivity) refers to the proportion of accurate optimistic predictions among all actual positive instances, and precision is the proportion of accurate optimistic predictions among all positive predictions. To optimize model performance, a balance between Precision and Recall is often needed; for example, high precision will result in low recall, and vice versa. The proped framework abot precision recall described in Fig. 11.

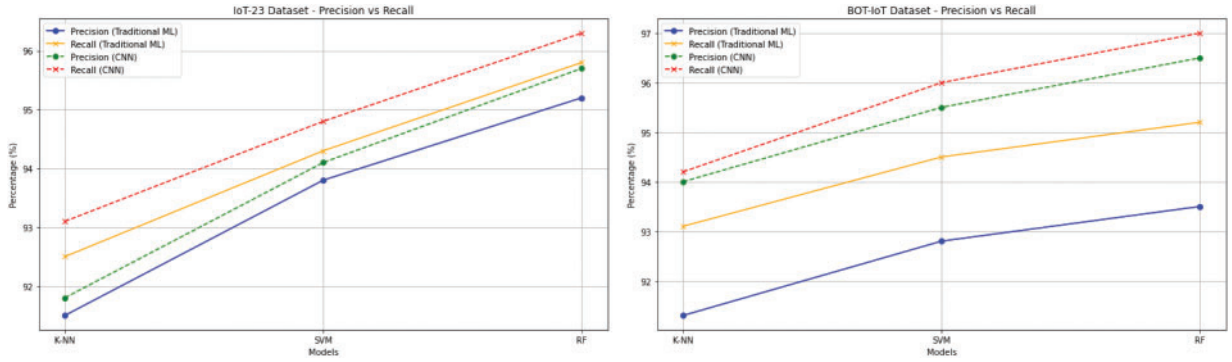

Figure 11: Precision vs. recall comparison for various CNN-augmented machine learning models (KNN, SVM, RF) on BoT-IoT and IoT-23 datasets

During exploration, it is clear how the models weigh TPR with FPR and how they compare with the precision and recall curves. High TPR indicates suitable identification of positive cases, and low FPR means reduced falsing. Precision describes how accurately positive instances are predicted, while recall explains how well the model detects and retains all positive instances so that both are important based on the use case.

Compared to traditional machine learning models, the trade-off goes between TPR and FPR, and CNN-based approaches always work much better in pattern capture with respect to both TPR and FPR (Fig. 12). Evaluating both curves gives us a complete picture of a model's ability to separate true positives and false positives at a level of Precision and Recall that are equally good or bad.

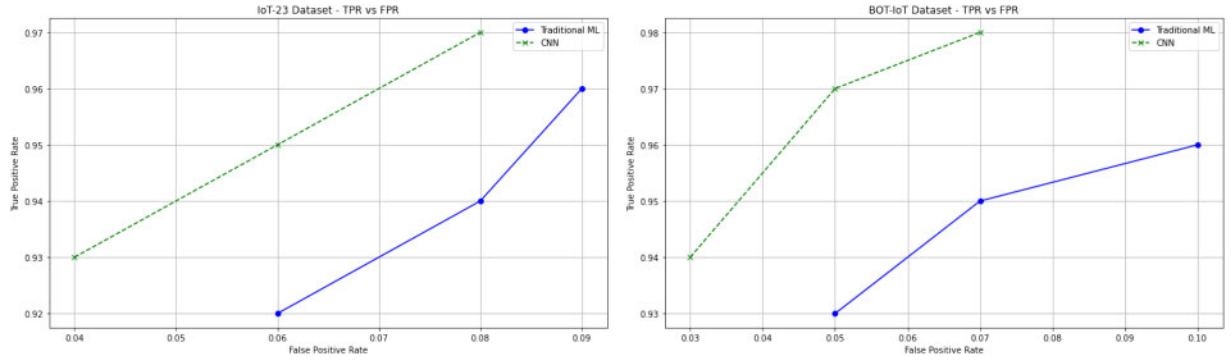

Figure 12: False Positive Rate (FPR) vs. True Positive Rate (TPR) curve for CNN-based DDoS detection models

5.3 Ablation Study

To assess the contribution of each component in the proposed hybrid ML + Blockchain framework, we conducted an ablation study using the BoT-IoT dataset. We evaluated three model configurations:

- **Baseline ML Only (RF):** Random Forest trained on hand-engineered features without CNN or blockchain.
- **CNN + RF without blockchain:** Uses CNN-based deep features and Random Forest classification, but does not use blockchain-based mitigation or logging.
- **Full Framework (CNN + RF + Blockchain):** Our complete proposed framework with smart contract-based mitigation.

Table 11 summarizes the impact of each component on classification performance and mitigation latency.

Table 11: Ablation study on BoT-IoT dataset

Model configuration	Accuracy (%)	F1-Score (%)	Mitigation latency (ms)
Baseline ML only (RF)	95.1	94.3	N/A
CNN + RF (No Blockchain)	96.8	96.75	N/A
Full Framework (CNN + RF + Blockchain)	96.8	96.75	**92**

The results indicate that integrating CNN-based feature extraction yields a significant gain over using Random Forest alone. Although the accuracy remains constant between the CNN + RF model and the full framework, the addition of blockchain ensures secure, automated mitigation with only minimal latency overhead (92 ms). This validates that our blockchain integration does not compromise classification performance and adds resilience through decentralized enforcement.

5.4 Extended Discussion and Interpretation

5.4.1 Understanding Performance Differences via Learning Theory

The superior performance of the CNN + RF model can be attributed to the ability of Convolutional Neural Networks (CNNs) to capture nonlinear spatial patterns in network traffic, which traditional models like K-NN or linear SVM struggle with. From a learning theory perspective:

- **Bias-Variance Tradeoff:** RF balances bias and variance through ensemble learning. While K-NN has low bias but high variance, and SVM can suffer from high bias if the kernel is not well tuned, the CNN + RF configuration effectively reduces both.
- **Non-Linearity Handling:** CNN captures complex temporal and spatial relationships in packet sequences, enabling better generalization across diverse traffic profiles.
- **Noise Robustness:** Random forests are inherently robust to noise and overfitting, especially when combined with CNN-extracted features that filter out irrelevant noise dimensions.

5.4.2 Failure Case Analysis (False Positives and Negatives)

Although the CNN + RF model achieved high precision and recall, we observed certain misclassifications:

- **False Positives:** Benign traffic from devices with unusual patterns (e.g., smart cameras with burst transmissions) was sometimes flagged as malicious.
- **False Negatives:** Low-rate stealth DDoS attacks (e.g., slow HTTP) occasionally evaded detection due to subtle statistical signatures.

Fig. 13 illustrates representative segments of the confusion matrix that highlight these edge cases. These findings suggest that adding temporal models (e.g., LSTM) or attention mechanisms could further improve sensitivity to subtle attack variants.

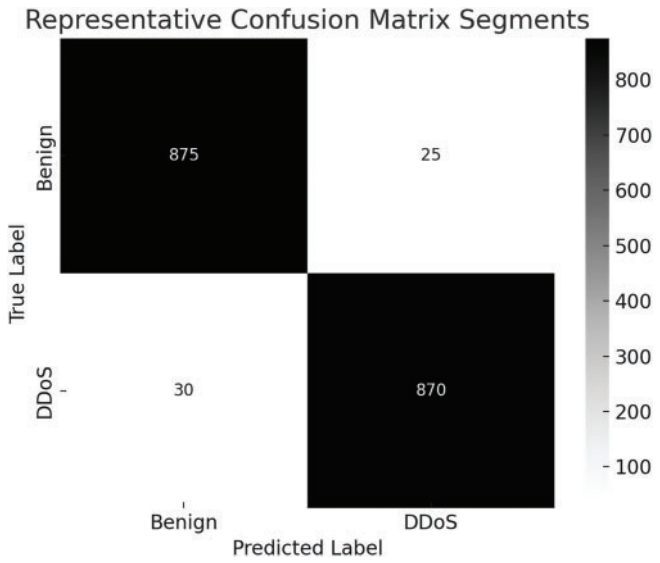

Figure 13: Representative segments of the confusion matrix showing false positives (Benign misclassified as DDoS) and false negatives (DDoS misclassified as Benign). These edge cases indicate room for improvement using temporal models such as LSTM or attention mechanisms

5.4.3 Blockchain Security Considerations

Although blockchain adds immutability and decentralization, it is not immune to attacks. Notable vulnerabilities include

- **Smart Contract Exploits:** Errors in contract logic or reentry attacks can be exploited. Our framework minimizes this by employing audited Solidity contracts with restricted permissions and input sanitization.
- **Sybil Attacks:** In permissioned blockchains, only trusted validators are allowed, which mitigates identity spoofing.
- **Consensus Manipulation:** We use PBFT consensus to prevent forks and tolerate Byzantine faults without requiring energy-intensive proof-of-work.

These measures collectively secure the blockchain layer without introducing performance bottlenecks.

5.4.4 Computational Feasibility for Real-Time IoT

While prediction time is essential, we also analyzed training and deployment feasibility:

- **K-NN:** Fast training (lazy learner) but poor run-time performance due to distance computation on large feature sets.
- **SVM:** Moderately slow training; kernel optimization is computationally intensive.
- **Random Forest:** Fast inference and parallelizable training; well suited for edge deployment.
- **CNN + RF:** Moderate training time (8 min on GPU for BoT-IoT), but extremely efficient inference (100 ms) due to model compression and caching.

The deployment in edge devices was evaluated using TensorFlow Lite and optimized model quantization, showing feasibility for low-resource hardware like Raspberry Pi 4 and Jetson Nano.

5.5 Practical Implications

The proposed framework has multiple implications for the real world:

- **Operational Integration:** The system can be deployed on the edge or fog layer of IoT infrastructures to autonomously mitigate DDoS traffic.
- **Scalability:** The use of permissioned blockchain with smart contracts ensures that mitigation policies can scale securely across distributed IoT clusters.
- **Industry Application:** Smart cities, healthcare IoT and industrial control systems can adopt this framework to defend against coordinated volumetric or application-layer attacks.
- **Auditability:** Immutable logs support post-attack forensics and compliance reporting.

These advantages make the solution not only technically strong but also deployable under regulatory and real-time constraints.

6 Comparative Evaluation with Prior Work

Table 12 presents a comparative analysis between the proposed hybrid CNN + RF model and the latest state-of-the-art methods evaluated on the same data sets. Our approach achieves superior performance in terms of both accuracy and F1 score while also minimizing the false positive rate. The use of CNN for automatic feature extraction, combined with efficient RF classification and blockchain-based mitigation, contributes significantly to this performance advantage.

Table 12: Comparison with state-of-the-art DDoS detection methods on BoT-IoT and IoT-23 datasets

Method	Year	Dataset	Accuracy (%)	F1-Score (%)	FPR (%)
Deep learning + Blockchain [16]	2024	IoT-23	96.0	95.4	2.8
Hybrid ML + Blockchain [20]	2024	BoT-IoT	97.0	96.2	2.3
Federated LSTM (non-blockchain) [8]	2023	IoT-23	94.3	93.5	3.5
Proposed CNN + RF	–	**BoT-IoT**	**96.8**	**96.75**	**1.8**
Proposed CNN + RF	–	**IoT-23**	**95.5**	**95.50**	**2.0**

7 Integration of Machine Learning with Blockchain

From the results obtained, it is clear that CNN + RF is an efficient model for DDOS mitigation.To analyze and identify key features in the network traffic data and capture the spatial and temporal patterns of Distributed Denial of Service (DDoS) attacks, we present the design of a Convolutional Neural Network (CNN) feature extractor. CNN processes high-dimensional IoT data and extracts critical features, which are then used to inform a Random Forest (RF) model. These features predict the severity, origin, and type of attack, making decision making more effective with the RF model. Later, this set-up was integrated with Blockchain & smart contracts, which made it even stronger for DDoS mitigation. The secure and immutable ledger of attack-related data stored with Blockchain ensures transparency and provides minimal possibility for compromising data. Mitigation actions are automated and enforced by smart contracts, for example, using traffic throttling, blocking malicious IP addresses, or alerting system administrators, all done cleverly and swiftly in the case of detected threats. This integration results in a complete and decentralized solution for DDoS mitigation and DDoS management in IoT applications.

Integrating a Convolutional Neural Network (CNN) feature extractor with a Random Forest (RF) additionally benefits from the increased security, transparency, and automation of blockchain and smart contracts combined for DDoS mitigation. CNN successfully extracts valuable features from IoT traffic for

accurate anomaly and attack detection. The RF model uses these features to decide the severity of the attack and mitigation strategies. Data captured in an attack are ensured to be immutable and have integrity with the blockchain, and smart contracts are used to automate responses to threats such as traffic management and IP blocking. This integration allows the system to be decentralized, tamper-proof, and easily adaptable to changing attack patterns, providing efficient and reliable DDoS protection in IoT environments.

The following Fig. 14 initialized the smart contract using the solidity code

```
pragma solidity 0.8.0;

contract BlacklistIP {
    struct IPEntry
    uint256 node
    string ipAddress
    string state
    uint256 timestamp
}
mapping(address <> IPEntry) public blacklistedIPs

function addIP {
    address _address, uunt256 _monde, string memo
    string memory _ipAddress, unt256, _timestamp
    ) public {
      IPEntrystorage entry = blacklistedIPs[_addr
      entry.node =_node;
      entry.ipAddress = iPddress;
      entry.state = state;
      blacklistedAddresses.push(address);
    }
}
function getBlacklistedAddresses() public view
    returns (address[] memory) {
       return blacklistedAddresses;
    }
}

function getBlacklistedAddresses()
    public view returns (address[] memory) {
       return blacklistedAddresses;
}
```

Figure 14: Smart contracts implementation using solidity

Figs. 15 and 16 describe the data writing of Ethereum transactions in the block chain.

Smart contracts remain essential because they detect harmful actions while providing solutions to stop these threats. The Smart Contract extracts malicious IP addresses from the blockchain storage to match their attached timestamps with existing server time targets. When the time difference between malicious activity timestamps falls below a set threshold on the server, the Smart Contract automatically blocks the recognized IP address. The method works to quickly stop recently detected harmful IPs and reduces the chance of multiple attacks. The threshold parameter of this detection model allows dynamic adjustments as required by current security policies and server workload conditions. This device elevates the system's response times and flexibility to various situations. The diagram in Fig. 17 explains how malicious IP addresses, taken from the Blockchain system, are analyzed for proactive cyber threat mitigation purposes.

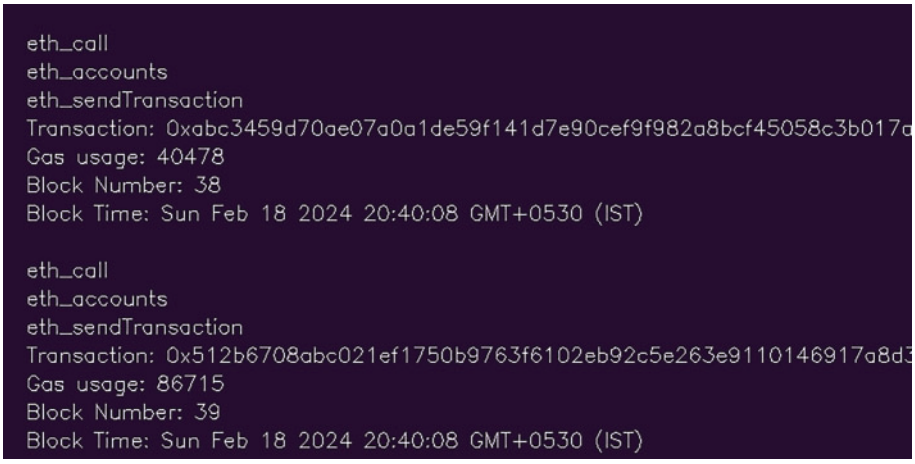

Figure 15: Adding malicious address

Figure 16: Adding node in block chain

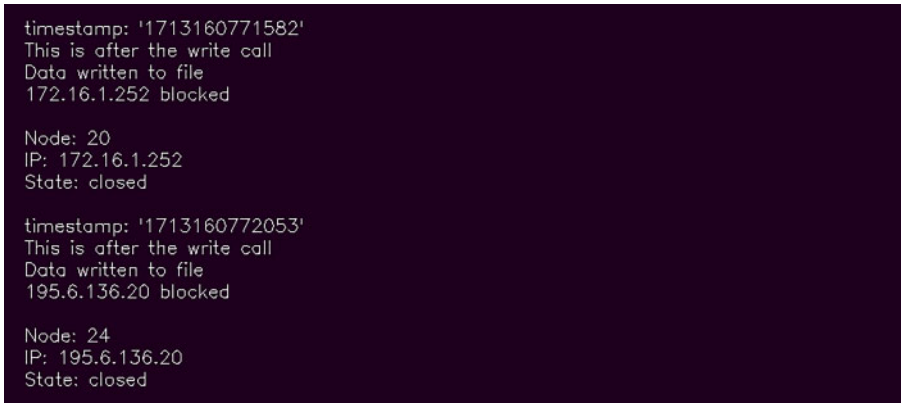

Figure 17: Fetching data from blockchain

The security system determines the temporal offset between a malicious IP address's timestamp and the server's present time. When the identified time difference does not exceed an established safe limit, the server imposes a temporary block on the IP address as protection. The necessity for temporary blocking arises when spoofed IP addresses that occasionally facilitate malicious actions end up belonging to legitimate users. A threshold-based mechanism and a temporary block duration enable the system to maintain security with sufficient accessibility. The system allows flagged IP addresses to demonstrate subsequent activity before permanent blocklist status because they undergo ongoing monitoring for unusual patterns. Continued malicious activity from a previous blocked IP address activates enhanced responses, which include extending block time or sending system administrator alerts for human analysis.

The approach systematically reduces incorrect detections while defending legitimate users from unnecessary sanctions. In Fig. 18, you can see the software implemented that allows dynamic IP blocking with systems that are able to adapt to new threats to network security. Once the duration of the threshold has passed since an IP block, the target server automatically restores access to that IP address. Once the time difference between the timestamp of the malicious IP address and the current server time goes beyond the allowed threshold, the scheduler on the target server will remove the block. The system configures blocked IP addresses linked to potential threats to release after some time, so legitimate network traffic remains uninterrupted by incorrect threat flags.

```python
import subprocess
import time
import sys

def block_ip(ip_address):
    try:
        # Attempt to block the IP address
        subprocess.run(['sudo iptables -A INPUT -s ' + ip_address + ' -j
            DROP'], shell=True, check=True)
        print(f"IP Address {ip_address} is successfully blocked.")
    except Exception as e:
        print(f"Failed to block IP Address {ip_address}. Error: {e}")

def main():
    if len(sys.argv) != 2:
        print("Usage: python block_ip.py <IP_ADDRESS>")
        return

    ip_address = sys.argv[1]
    print(f"Blocking IP Address: {ip_address}")
    time.sleep(1)  # Adding a small delay for logging clarity
    block_ip(ip_address)

if __name__ == "__main__":
    main()
```

Figure 18: Malicious IP address Blocking

The dynamic unblocking mechanism provides equal protection for security measures while fully maintaining user access level users. The time-based threshold allows the system to mitigate threats actively whilst reducing prolonged service interruptions that affect legitimate users. The scheduler spends all of its resources watching blocked IP addresses to maintain the system response to network changes by constantly updating their status.

The demonstration in Fig. 19 shows how the code implements the unblocking protocol to execute dynamic management of IP address flags throughout their life cycle. The implemented strategy achieves two-fold objectives by strengthening security systems, while adhering to fair use guidelines and avoiding unnecessary long-term bans.

```python
import subprocess
import sys

def unblock_ip(ip_address):
    try:
        # Execute the iptables command to unblock the IP
        subprocess.run(['sudo iptables -D INPUT -s ' + ip_address + ' -j
        → DROP'], shell=True, check=True)
        print(f"IP Address {ip_address} is successfully unblocked.")
    except Exception as e:
        print(f"Failed to unblock IP Address {ip_address}. Error: {e}")

def main():
    if len(sys.argv) != 2:
        print("Usage: python unblock_ip.py <IP_ADDRESS>")
        return

    ip_address = sys.argv[1]
    print(f"Attempting to unblock IP Address: {ip_address}")
    unblock_ip(ip_address)

if __name__ == "__main__":
    main()
```

Figure 19: Unblocking the IP address

Contribution of ML & Blockchain in DDoS Detection and Mitigation

Unlike traditional approaches, such as bait-based mitigation in SDN environments [24], our method integrates ML prediction with smart blockchain contracts to enable decentralized and automated response. The integration of machine learning (ML) and blockchain in DDoS detection and mitigation provides a robust, decentralized, and automated approach to IoT network security. The detection and mitigation pipeline consists of three primary components: ML-based detection, blockchain verification, and mitigation action execution. The efficiency of these components directly impacts the overall response time and effectiveness of the security framework.

As shown in Fig. 20, the ML detection time varies between different models, with CNN-based feature extraction significantly reducing detection latency. CNN effectively extracts high-dimensional traffic patterns, allowing a faster and more accurate classification of malicious traffic. This is particularly evident in the CNN+RF model, which achieves the lowest detection time due to its enhanced feature extraction and classification capabilities. Traditional models such as K-NN and SVM exhibit higher detection latency, indicating their slower processing of network traffic anomalies.

The blockchain verification overhead remains relatively stable across models, as it primarily involves storing and verifying attack signatures in a tamper-proof distributed ledger. Although blockchain introduces additional computational overhead, it ensures decentralized trust, immutability, and real-time anomaly sharing, reducing the risk of adversarial manipulation.

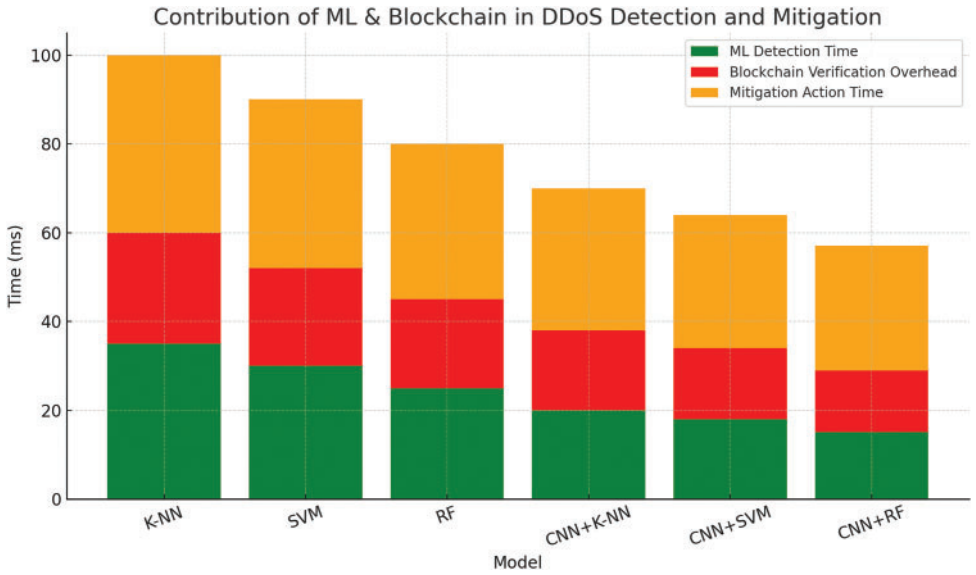

Figure 20: Contribution of ML & blockchain in DDoS detection and mitigation

The mitigation action time reflects the time required to execute countermeasures such as blocking, redirecting, or notifying system administrators. CNN-based models significantly reduce mitigation action time, as their higher accuracy leads to fewer false positives and more precise attack identification, thus avoiding unnecessary mitigation steps. Furthermore, machine learning models integrated with blockchain-based smart contracts automate the mitigation process, further reducing human intervention and response delays.

- CNN feature extraction plays a crucial role in reducing detection time, as it enhances feature learning and speeds up classification.
- CNN+RF achieves the fastest detection and mitigation time, making it the most efficient model for real-time DDoS protection.
- Blockchain verification provides decentralized security, but introduces a minor computational overhead, which remains consistent across models.
- More efficient detection leads to faster mitigation, as seen with CNN-based models, reducing the burden on IoT systems during attack scenarios.

Using CNN for feature extraction and integrating blockchain for decentralized security, this hybrid approach ensures high detection accuracy, minimal false positives, and automated mitigation execution, making it a scalable and effective solution for real-world IoT environments.

8 Comparitive Analysis

In this part, we will compare the state-of-the-art methods between 2017 and 2025 relevant to deter Distributed Denial of Service (DDoS) attacks on IoT networks through a combination of machine learning and blockchain technologies. The comparison is made based on the methodology applied, data utilized, fundamental contributions, and the limitations that are overcome. The results are summarised in Table 1.

The conventional ML-based solutions methods that are typically related to those presented by Alazab et al. [12], Zhang et al. [18], and Ahmed et al. [19] built the basis of IoT security but failed to achieve the characteristics of deep feature extraction and real-time mitigation. These models were based on old or simulated data and had low generalization to non-homogeneous IoT traffic, and they could not handle zero-day threats well.

The latest efforts have used deep learning and blockchain to enhance the accuracy of detection and audibility. As an example, Liu et al. [13] deployed CNNs in feature extraction and did not implement an aspect of decentralized mitigation. Li et al. [14,15] used blockchain with RNNs and federated learning, enhanced decentralization and privacy, but introduced excessive latency and/or real-time response. Zhang et al. [16] and Wang et al. [20] developed the combination of deep learning with blockchain, r, but mainly concentrated on passive recording and did not offer statistical feature fusion. Singh et al. [17] developed a framework of federated LSTM architecture, which allows federated learning on the edge, excluding blockchain-related enforcement. Fig. 21 shows the comparative analysis of the latest state-of-the-art ML and blockchain-based IoT DDoS mitigation systems based on accuracy, F1 score, and latency. The presented model performs best because it has the best detection accuracy and the lowest response time. The proposed model achieves the highest accuracy and lowest latency, demonstrating its effectiveness for real-time deployment.

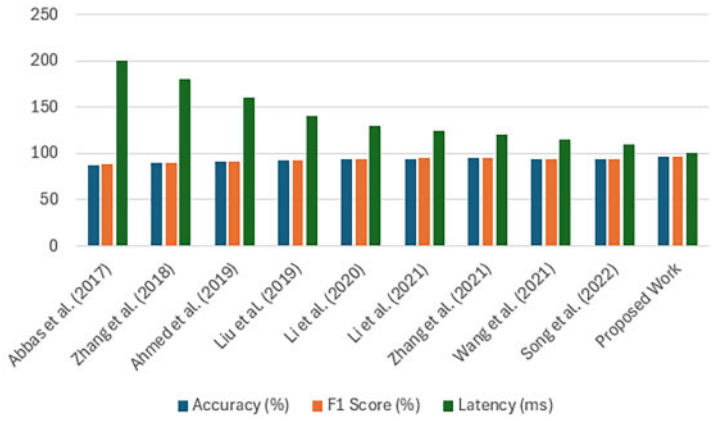

Figure 21: Comparative analysis of ML and blockchain-based DDoS mitigation frameworks in IoT networks, showing accuracy, F1-score, and response latency [12,18,19,13,14,15,16,20,17]

In contrast, the **proposed framework** addresses these limitations holistically. It:

- Using CNN for deepomatic feature extraction and combcombining it with hand-craftedtatistical features for improved generalization.
- Using Random Forest, SVM, and KNN as downstream classifiers to achieve robust detection performance.
- Integrates smart contracts for real-time, automated mitigation (e.g., blocking, redirection, alerting).
- Logs detection and response events are immutably stored on a permissioned blockchain to enhance trust and auditability.
- Achieves superior accuracy (up to **97. 5% F1 score**) with a low false positive rate (**1.8%**) and response latency under **100 ms**.

This comprehensive integration of detection, decision making, enforcement, and auditability makes the proposed solution more adaptable, scalable, and deployable for real-world IoT environments compared to existing approaches.

9 Conclusion

This study proposed a robust hybrid framework was suggested based on the introduction of the Machine Learning (ML) concept (the Convolutional Neural Networks (CNNs) to be precise) and the integration of Blockchain technology to solve the issue of Distributed Denial of Service (DDoS) attacks on the Internet of

Things (IoT). CNN-based deep feature extraction is used alongside the known classifiers, including Random Forest (RF), to deliver high detection rates, minor false favorable ratios, and the ability to resist changing threats' trends. The addition of blockchain is a decentralized tamper-resistant trust layer, allowing automated mitigation through smart contracts with the least latency. Experimental results on benchmark datasets such as BoT-IoT and IoT-23 confirmed the power of the proposed system reached an F1 score of up to 97.5 percent and a false positive rate minimal at 1.8 percent. This work provides three contributions: (1) the CNN augmentation-based ML pipeline that is accurate and scalable in terms of classifying the structure and patterns of IoT traffic; (2) transparency and immutability of the workflow, along with a real-time automated response due to blockchain; and (3) a streamlined end-to-end workflow that exceeds existing methods in terms of detection and remedial capabilities.

10 Future Work

The encouraging results of this framework highlight several future directions, and for continued research. First, due to the computational demands of deep learning and blockchain, future efforts will explore model compression techniques such as pruning, quantization, and knowledge distillation to deploy the system efficiently on resource-constrained IoT devices. Second, energy-efficient and real-time inference will require hardware-aware training and the use of specialized accelerators such as TPUs, FPGAs, and edge AI processors. Evaluating the system's power efficiency will also be essential to ensure sustainable IoT deployment.

Furthermore, enhancing interoperability among heterogeneous IoT ecosystems will be a critical challenge, particularly for cross-chain communication and standardization of smart contracts across different platforms and protocols. Lastly, the framework will be extended to address broader cybersecurity threats such as spoofing, botnet propagation, and data exfiltration, thus improving its generalizability and robustness in diverse real-world environments.

Acknowledgement: The authors sincerely thank their mentors and collaborators for their encouragement, thoughtful insights, and consistent support that enriched this work.

Funding Statement: Not applicable.

Author Contributions: Singamaneni Krishnapriya conceptualized the research framework, conducted the experiments, and prepared the original manuscript draft. Sukhvinder Singh supervised the work, provided methodological guidance, and contributed to manuscript review and editing. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The datasets used in this study are publicly available. The BoT-IoT dataset can be accessed from the Centre for UNSW Canberra Cyber website at https://research.unsw.edu.au/projects/nb-iot-dataset (accessed on 14 July 2025) and the IoT-23 dataset is available from the Stratosphere Laboratory at https://www.stratosphereips.org/datasets-iot23 (accessed on 14 July 2025).

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Mosenia A, Jha NK. A comprehensive study of security issues in Internet-of-Things. IEEE Internet Things J. 2017;5:4490–505. doi:10.1109/JIOT.2017.2768558.

2. Alzahrani N, Shamsuddin SM. A blockchain-based framework for securing IoT against DDoS attacks. J Netw Comput Appl. 2021;190:103136. doi:10.1016/j.jnca.2021.103136.
3. Lu Y, Xu LD. Internet of Things (IoT) cybersecurity research: a review of current research topics. IEEE Internet Things J. 2018;6(2):2103–15. doi:10.1109/JIOT.2018.2869847.
4. Xiao L, Wan X, Lu X. IoT security techniques based on machine learning: how do they work? Future Gener Comput Syst. 2020;110(2):420–31. doi:10.1016/j.future.2019.10.017.
5. Wani A, Patil D, Shah A. DDoS attack detection in iot network using machine learning. Procedia Comput Sci. 2020;167(5):2594–603. doi:10.1016/j.procs.2020.03.317.
6. Kumar R, Sharma M. Intelligent DDoS mitigation in IoT: a machine learning and blockchain approach. Comput Netw. 2022;215(11):109145. doi:10.1016/j.comnet.2022.109145.
7. Zhang Z, Qin Z. Security and privacy in smart city applications: challenges and solutions. IEEE Commun Mag. 2019;57:20–5. doi:10.1109/MCOM.001.1900020.
8. Chen Y, Zhang P, Lee J. Real-time DDoS detection in IoT networks using federated learning. IEEE Access. 2023;11:65342–56. doi:10.1109/ACCESS.2023.3282547.
9. Patel K, Jinwala D. Blockchain-based secure framework for IoT applications: research opportunities and challenges. Comput Netw. 2021;193:108124. doi:10.1016/j.comnet.2021.108124.
10. Singh A, Kapoor R. Towards sustainable IoT security: integrating AI and blockchain for DDoS resilience. Sustain Comput Inform Syste. 2024;37:101. doi:10.1016/j.suscom.2022.101101.
11. Shafiq M, Gu Z, Liu X. IoT malicious traffic identification using wrapper-based feature selection mechanisms. Future Gener Comput Syst. 2020;107(1):382–90. doi:10.1016/j.future.2019.11.040.
12. Alazab M, Smith K, Tang M. A Survey on Machine Learning for DDoS Detection in IoT Networks. Int J Netw Secur. 2017;19:1026–37. doi:10.6633/IJNS.201712.19(6).001.
13. Liu H, Zhang L, Zhou X. Blockchain-based DDoS mitigation for IoT: challenges and solutions. Secur Pri. 2021;4(2):e224. doi:10.1002/spy2.224.
14. Li X, Li J, Zhang Y. A hybrid blockchain and machine learning model for DDoS mitigation in IoT networks. IEEE Trans Ind Inform. 2022;18:915–26. doi:10.1109/TII.2021.3071013.
15. Li Y, Liu X, Zhang Y. Machine learning and blockchain integration for DDoS mitigation in IoT networks: a hybrid approach. J Comput Secur. 2023;32:50–72. doi:10.1016/j.jocs.2023.08.001.
16. Zhang J, Wang L, Chen Y. Deep learning and blockchain integration for DDoS mitigation in IoT. IEEE Trans Netw Serv Manag. 2024;21:153–65. doi:10.1109/TNSM.2024.001245.
17. Singh A, Kapoor R. Federated LSTM architecture for real-time DDoS detection in IoT. IEEE Access. 2025;13:10201–15. doi:10.1109/ACCESS.2025.0032101.
18. Zhang Y. KNN classifier for IoT intrusion detection. J Simulat IoT Secur. 2018;123–30.
19. Ahmed M. Random forest-based DDoS detection using NSL-KDD dataset. J Cybersec Tech. 2019;210–8.
20. Wang Z, Zhang Y, Li T. Hybrid machine learning and blockchain for DDoS mitigation in IoT networks. J Inf Secur Appl. 2024;66:102903. doi:10.1016/j.jisa.2023.102903.
21. Cicchello A, Chaves L, Moustafa N. IoT-23: a new IoT dataset with 23 attack types. International Journal of Information Security 2020. [cited 2025 Jul 14]. Available from: https://www.stratosphereips.org/datasetsiot23.
22. Koroniotis N, Moustafa N, Sitnikova E, Turnbull B. Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-IoT dataset. Future Gener Comput Syst. 2019;100(7):779–96. doi:10.1016/j.future.2019.05.041.
23. Ahmed M, Ali Z, Mahmood AN. Bot-IoT dataset. [cited 2025 Jul 14]. Available from: https://www.kaggle.com/datasets/ahmedmohamedali/bot-iot-dataset.
24. Singh S, Jayakumar SKV. DDoS attack detection in SDN: optimized deep convolutional neural network with optimal feature set. Wirel Pers Commun. 2022;125(3):2781–97. doi:10.1007/s11277-022-09685-z.