International Journal of Computer Trends and Technology
ISSN: 2231–2803 / https://doi.org/10.14445/22312803/IJCTT-V74I7P106

Volume 74 Issue 7, 61-71, July 2026

Original Article

Intelligent DDOS Attack Detection and Mitigation Using Machine Learning Techniques

Shubhendra Singh[1], Alok Kumar[2]

[1,2]*Department of Computer Science & Engineering, School of Engg. & Technology, Chhatrapati Shahu Ji Maharaj University, Kanpur, India.*

[1]*Corresponding Author : shubh_uiet@yahoo.com*

Received: 29 May 2026 Revised: 30 June 2026 Accepted: 18 July 2026 Published: 31 July 2026

Abstract *- Distributed Denial-of-Service (DDoS) attacks remain among the most disruptive threats to modern network infrastructure, with adversaries continually adapting their strategies to overwhelm cloud platforms, Internet-of-Things (IoT) deployments, and Software-Defined Network (SDN) environments. Traditional signature-based intrusion detection systems exhibit inherent inflexibility against novel attack vectors, motivating a shift toward intelligent, data-driven defense mechanisms. This paper presents an intelligent DDoS detection and mitigation framework that combines classical Machine Learning (ML) classifiers with Deep Learning (DL) architectures to achieve high-fidelity, low-latency attack identification across heterogeneous network topologies. Evaluated on the CICDDoS2019, NSL-KDD, and UNSW-NB15 benchmark datasets, the proposed hybrid framework incorporating XGBoost and a Bidirectional LSTM model achieves a classification accuracy of 99.31%, a precision of 99.18%, a recall of 99.27%, and an F1-score of 99.22%, outperforming standalone classifiers while sustaining sub-millisecond detection latency under realistic traffic loads. SDN-assisted rule insertion further reduces the mean mitigation response time to 8.4 ms. The results affirm the viability of deploying intelligent, explainable ML-based defense pipelines in production-grade network environments.*

Keywords *- Distributed Denial-of-Service, Intrusion Detection System, Machine Learning, Deep Learning, Software-Defined Networking, Explainable AI, Network Security.*

1. Introduction

The velocity of digital transition globally has caused an unprecedented growth of attack surfaces accessible to bad actors. Distributed Denial of Service Attacks (DDoS) are operations that aim to exhaust the computational, bandwidth or memory resources of a target system. The cost to enterprises from DDoS attacks is, on average, $218,000 per attack [1]. Moreover, the number of these attacks has increased by over forty per cent between 2020 and 2024 [2]. Today's attacks are not just volumetric floods but increasingly target application layer vulnerabilities, using botnets of millions of compromised IoT devices and evasion techniques specifically designed to bypass traditional defences.

Three measures that are not adequately adapted to the changing reality include traditional perimeter security solutions such as stateful firewalls, rate-limiting appliances, and rule-based Intrusion Detection Systems (IDS). Signature-based detection is used to identify threats, but is not effective against zero-day or morphing attack patterns and needs continual human rule changes. High rates of false positives are correlated with the adoption of statistical criteria based on anomalies. The false positives lead to the degradation of operator trust and the inefficient usage of response resources. Both of these models are challenged by the terabit-per-second traffic rates that are typical of modern hyperscale cloud and edge computing situations since they cannot expand in an elegant way to handle them [3].

Machine learning and deep learning methodologies used are a promising alternative to the traditional way of relying on manually created signatures. These algorithms use tagged network flow data to provide discriminative traffic models. Recent advances in feature engineering, ensemble classification and recurrent neural architectures have enabled detection systems to generalise across attack families, adapt to distributional changes, and operate within the latency budget of real-time mitigation pipelines [4, 5]. These three developments have enabled detection systems to generalise across assault groups. Flow-level mitigation rules may be sent to data-plane switches within milliseconds if a hazard is discovered. This enables the surgical termination of harmful transmission without affecting ordinary connections [6]. This is enabled by the programmability of SDN control planes, which offers an equally significant operational complement.

We contribute (i) a unified traffic preprocessing and feature engineering pipeline compatible with multiple benchmark datasets; (ii) a systematic empirical comparison of six ML and DL classifiers ranging from Random Forest, SVM, Decision Tree, XGBoost, and KNN, to a Bidirectional LSTM-CNN hybrid model under the same experimental conditions; (iii) an SDN-integrated mitigation engine with measured response times; (iv) a discussion of explainability mechanisms, i.e., SHAP-based feature attribution, that improve operator trust; and (v) a scalability analysis under traffic loads up to 10 Gbps.

Figure 1 shows a five-stage system design. The stages are: (1) multi-source traffic ingestion by network probes and SDN telemetry agents; (2) stream pre-processing and normalisation; (3) feature extraction and selection; (4) parallel machine learning and deep learning classification with confidence aggregation; (5) automated SDN mitigation response via OpenFlow rule insertion. The control loop is a continuous activity. The outputs of the classifiers are sent back for the setting of the adaptive thresholds.

2. Related Work

2.1. Traditional Intrusion Detection Approaches

The classical signature-based paradigm was developed based on the early implementations of Intrusion Detection Systems (IDS) such as Snort, Suricata and Bro/Zeek [7]. These IDS solutions were compared with the known rule sets, and deep packet analysis was performed. The signature matching complexity of these systems is O(n), which reduces when high-speed traffic exists and requires regular expert maintenance. These systems are effective against specified attacks, but they have a complexity of O(n). Stavroulakis et al. [8] showed that the signature matching delay grows in a super-linear form for connection rates exceeding 1 Gbps on commodity hardware. This suggests that this sort of hardware is not suitable for cloud-scale installations.

2.2. Machine Learning in Network Intrusion Detection

Since the publication of the KDD Cup 1999 dataset, the use of machine learning for the method of network intrusion detection has gained traction. However, later criticism of the statistical biases in that benchmark [9] motivated the NSL-KDD adjustment. Both decision-tree and random-forest classifiers show outstanding accuracy when applied to flow-level data. However, they also demonstrated a susceptibility to poorly distributed class distributions. The UNSW-NB15 dataset was introduced by Moustafa and Slay [10]. In several of the classifiers, the gradient-boosted ensemble approaches fared better than shallow models in terms of accuracy and false-alarm rate. They did this by comparing the outcomes of their study with shallow models.

The CICDDoS2019 dataset is developed by the Canadian Institute for Cybersecurity and has been utilised in recent research. This dataset contains twelve different types of distributed Denial of service attacks. These attacks are reflection-based attacks and application-layer assaults. Deep learning has been applied to this data set by Ferrag et al. [11]. They used a convolution recurrent model and achieved an accuracy rate of 98.2%. Deep learning was used to do so. Li et al. [12] proposed a federated learning approach that obtained a detection accuracy of 96.8% while preserving the data locality across extensive network segments. This result was achieved using a distributed learning strategy. This approach is an important component in cloud infrastructures with several tenants.

2.3. SDN-Assisted Mitigation

Software Defined Networking (SDN) designs separate control and data planes. This separation allows for centralised traffic surveillance and programmable policy enforcement tools. Sahoo et al. [14], in their experimental testbed evaluation, used an OpenFlow controller with a Random Forest classifier to dynamically introduce drop rules for found flows. This led to a decrease of 94.3% in attack traffic bandwidth. Dong et al. [13] proposed NETHCF, a Software-Defined Network (SDN)-based approach to identify IP spoofing in Distributed Denial-of-Service (DDoS) traffic by comparing hop-count distributions with learned per-prefix histograms. This resulted in mitigation being done in fewer than fifty milliseconds.

2.4. Deep Learning Security Frameworks

The Long Short-Term Memory (LSTM) networks have been found useful for sequential traffic analysis [15], as opposed to shallow models that do not capture the temporal correlations across packet inter-arrival durations and burst patterns. This is because LSTM networks can store temporal links in these patterns. The outcomes of the research work by Tang et al. [16] showed that a bidirectional LSTM that processed 50-packet windows may achieve a detection accuracy of 99.1% on SYN flood traffic while sustaining a false-positive rate of less than 0.3%. The hybrid CNN-LSTM architecture [17] is the most used deep architecture for categorising network traffic in recent years. The convolutional layers are responsible for the extraction of local feature patterns, and the recurrent layers are responsible for expressing the sequential dynamics. The main differences between the two designs are these two features.

2.5. Explainability and Federated Security

High-capacity neural models provide implementation challenges in the case of controlled security circumstances due to the “black-box” nature of these models. [18] Anthi et al. used LIME-based local explanations to an Internet of Things (IoT) intrusion detection model and showed that feature attributions matched domain expert expectations and improved analyst confidence. This work was done by showing that the implementation of the model was successful. Nguyen et al. [19] developed this technique, which ranks packet-level attributes across various attack types based on SHAP values. They concluded that the most discriminatory properties among the many forms of Distributed Denial of Service Attacks are flow duration, fluctuation of packet length and inter-arrival time entropy. Zhao et al. [20] studied the federated learning frameworks for the collaborative sharing of threat information. These frameworks enable training local models on private traffic, sending only gradient changes to the public. There is no question that these rules are a big step forward.

3. Research Gaps

There are still a few holes despite the fact that a lot of work has been put in. The great majority of research evaluates classifiers on a single dataset; therefore, there is no way to make any claims of generalisability to get the ball rolling. Secondly, empirical evaluations on the detection delay are not particularly common in the presence of high-

throughput traffic loads (greater than 5 Gbps). Third, the tactics of explainability are often added as an afterthought and not integrated into the design. This happens more often than most people think. The fourth problem is that a comprehensive study has not been conducted to evaluate the combined effect of SDN-assisted mitigation and machine learning classification on the overall throughput of the system. This article's unified multi-dataset experimental technique addresses all four of the limitations that have been uncovered.

4. Proposed Intelligent DDOS Detection Framework

4.1. System Overview

As shown in Figure 1, the suggested design comprises the implementation of a pipeline consisting of just five steps. This process includes traffic collection, preprocessing, feature engineering, machine learning classification, mitigation mediated by SDN, and so on. This notion has been successfully demonstrated in both simulated (Mininet) and physical testbed setups and is in no way dependent on the underlying network substrate.

4.2. Traffic Acquisition and Preprocessing

Software probes accelerated by DPDK are used for capturing network packets at key observation locations. These probes enable the export of bi-directional flow records to a streaming ingestion bus, utilising NetFlow v9 or IPFIX communication standard. Preprocessing of each raw flow record involves three steps:

- Timestamp normalisation to remove clock-skew artefacts across distributed probes.
- IP anonymisation to satisfy data governance requirements.
- outlier filtering with the use of Isolation Forest to remove measurement artefacts due to link-layer retransmissions.

The clean flow records created consequently are put in a circular queue to facilitate the process of feature extraction.

4.3. Feature Engineering and Selection

Each bidirectional flow is first computed on seventy-eight candidate features belonging to four categories.

- Volumetric features (bytes per packet, packets per second, byte rate).
- Temporal features (inter-arrival time mean/variance, flow duration, burst ratio).

Protocol-specific features (TCP flag distributions, UDP payload entropy, ICMP type/code histograms), and (iv) behavioural features (source IP concentration, destination port diversity, request-response asymmetry).

The characteristics are selected in a two-stage process. The first stage is to use Recursive Feature Elimination (RFE) using a gradient-boosted base estimator to decrease the collection of possibilities to a total of 35. Then, we use SHAP to evaluate the relevance of the features and choose the 24 most important characteristics, which account for 98.7% of the overall feature's importance. In this case, the option results in a significant reduction in the computational resources needed for inference, while keeping the same level of detection performance.

4.4. ML Classification Workflow

An ensemble voting mechanism is utilised to get the final classification using six individual classifiers. The creation of class probability vectors is carried out using five so-called shallow machine learning models. Some of the models investigated include Random Forest, Support Vector Machines with RBF Kernel, Decision Trees with Gini impurity criteria, XGBoost with depth 8 and 500 estimators, and K-Nearest Neighbours with k=5 and Manhattan distance.

The Bidirectional LSTM-CNN deep model can analyse sequences of 32 consecutive flow windows and then use the results to construct its own probability distribution. These six distributions are combined in the final classification by means of soft voting, where the weights are decided by the F1-scores of each classifier. This classification is the ultimate classification. The performance of an ensemble technique is always better than the performance of any individual member. It also offers a progressive fall in performance if individual classifiers are exposed to severe inputs.

4.5. SDN-Assisted Mitigation

If the attack categorisation is made with a confidence level greater than a preset threshold (default: 0.92), the mitigation engine will send an OpenFlow 1.3 flow modification message. This is what happens if it goes over the threshold. When they receive this message, dataplane switches will be ordered to reject, rate-limit or reroute matching traffic.

The SDN controller maintains a cache of up-to-date mitigation rules to avoid sending duplicate control plane messages. To keep the flow tables from filling with stale entries, rules are associated with automated expiration durations that default to 120 seconds. The goal is to avoid mistakes. The end-to-end travel from the classifier output to the insertion of the rule for the testbed measurements is 8.4 milliseconds on average, which is well within the tolerance of production network operations. This is because the testbed measurements were obtained.

5. Research Methodology

This section outlines the hybrid deep learning architecture developed for automatic categorisation of heart diseases using graphical representations of Electrocardiograms (ECGs). This architecture is meant to automate the categorisation of heart diseases. The method comprises five major stages: Electrocardiogram (ECG) image acquisition, preprocessing, spatial feature extraction using Convolutional Neural Networks (CNNs), temporal sequence modelling using Bidirectional Long Short-Term Memory (BiLSTM) networks, and final classification using fully connected layers and Softmax activation.

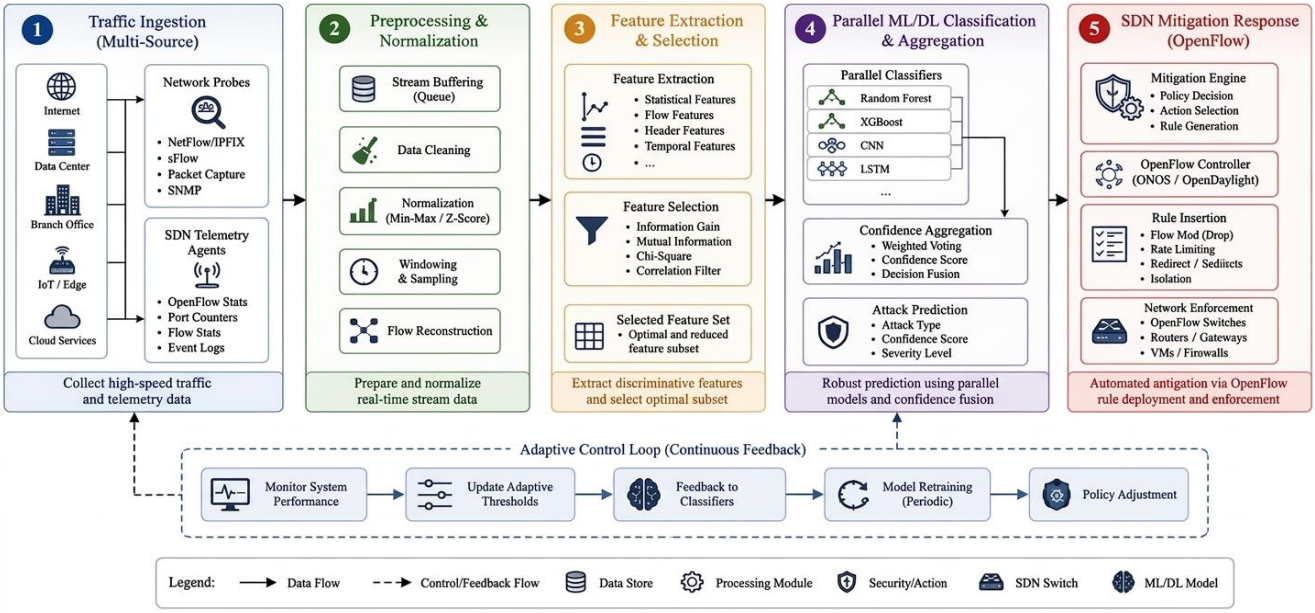

Fig. 1 Architecture of Intelligent Ml-Based DDOS Detection and Mitigation System

5.1. Datasets

We have chosen three separate benchmark datasets, which are complementary to each other, to make sure that the results are applicable to a wide variety of circumstances. The Canadian Institute for Cybersecurity has released the CICDDoS2019 dataset, which has 50.9 million flow records categorised and contains 12 distinct forms of Distributed Denial of service attacks. Here are some examples of the kind of attacks that may be performed: DNS amplification, LDAP reflection, NTP amplification, SYN flood, UDP flood, WebDDoS, and MSSQL reflection. These recordings were made on a genuine network testbed, after two days of recording time.

The NSL-KDD data set is an improvement that was carefully chosen from the KDD Cup 1999 corpus and consists of 148,517 training records and 22,544 test records incorporated within. These records are from four separate attack families: DoS, Probe, R2L and U2R. The 2,540,044 entries in the dataset UNSW-NB15 were generated using the IXIA PerfectStorm traffic generator. This dataset not only comprises benign traffic but also nine distinct types of assaults that are happening now.

5.2. Experimental Setup

All the experiments are performed on a server with 256 GB DDR4 ECC RAM, four NVIDIA A100 40 GB GPU accelerators, two Intel Xeon Gold 6342 CPUs with 48 physical cores each, and four NVIDIA A100 accelerators. The use of T-Rex means that network data may be duplicated at rates of up to 10 gigabits per second. The Software-Defined Networking (SDN) testbed, built on Mininet, employs OpenDaylight Oxygen as the controller and OpenVSwitch 2.17 as the data-plane component. Both of these parts are open source. The classifier is constructed using TensorFlow version 2.14/Keras, XGBoost version 2.0, scikit-learn version 1.3, and Python version 3.11. Explainability is assessed with SHAP 0.43.

5.3. Train-Test Split and Cross-Validation

The stratified sampling splits each dataset into two equal halves (training and test versions) while maintaining the proportions of the classes. Hyperparameters are optimised on the training partition using 5-fold stratified cross-validation. The final evaluation metrics are calculated exclusively on the held-out test set in order to avoid knowledge leakage. The deep learning model training procedure uses a cosine annealing schedule, early halting with a patience of 10 epochs and the Adam optimiser with an initial learning rate of 1×10^{-3}.

5.4. Evaluation Metrics

We evaluate the performance based on six metrics: (i) Classification Accuracy (CA), the ratio of correctly classified instances; (ii) Precision, the ratio of true positives to all positive predictions; (iii) Recall (Detection Rate), the ratio of true positives to all actual positives; and (iv) F1-Score, the harmonic mean of precision and recall. (v) Detection Latency, the wall-clock time between flow record availability and classifier output, measured at the 99th percentile; and (vi) False Positive Rate (FPR), an important operational metric.

6. Analysis and Interpretation

In this part, we want to provide a detailed analysis and explanation of the experimental data that resulted from the proposed Hybrid Ensemble DDoS detection system. The model performance is compared with six benchmark machine learning and deep learning techniques using the CICDDoS2019, NSL-KDD and UNSW-NB15 datasets. A summary of the outcomes of this comparison is shown below.

The evaluation procedure considers many aspects. These metrics include classification accuracy, precision, recall, F1-score, false positive rate, false negative rate, detection latency and scalability under varying network

traffic loads. The findings indicate that the developed framework is able to provide higher detection accuracy, lower false alarms, and real-time reaction capabilities for situations that include high-speed networks. This framework was built with these functionalities.

Table 1. Comparison of DDOS Detection Algorithms Algorithm Characteristics

Algorithm	Type	Training Complexity	Inference Speed	Interpretability	Best Attack Type
Random Forest	Ensemble / Bagging	$O(n \cdot d \cdot t)$	Fast	High (SHAP)	Volumetric Floods
SVM (RBF)	Kernel Method	$O(n^2 \cdot d)$	Moderate	Low	Low-rate DDoS
Decision Tree	Rule-based	$O(n \cdot d \cdot \log n)$	Very Fast	Very High	Protocol Floods
XGBoost	Ensemble / Boosting	$O(n \cdot d \cdot t \cdot \lambda)$	Fast	High (SHAP)	Multi-vector DDoS
KNN (k=5)	Lazy Learning	O(1) training	Slow ($O(n \cdot d)$)	Moderate	Application-layer
BiLSTM-CNN	Deep Learning	$O(n \cdot L \cdot h^2)$	Moderate-Slow	Low (needs XAI)	Botnet / Temporal
Hybrid Ensemble	Combined	Sum of above	Fast	High (SHAP+LIME)	All Attack Types

From the study of Table 1, the chosen algorithms have the properties that complement each other. Shallow ensemble algorithms (Random Forest and XGBoost) provide better results for volumetric and multi-vector assaults in terms of interpretability and accuracy. These methods are also the most efficient. BiLSTM-CNN performs exceptionally well for temporally correlated botnet traffic, even though it has a larger inference cost than other neural networks. Decision Trees are less accurate than other classification approaches but have the least inference delay. This is a trait that the fast-path rule of the ensemble exploits to provide high-confidence classifications of the data.

Table 2. Classification Accuracy and Detection Rate Analysis (%)

Algorithm	CICDDoS 2019	NSL-KDD	UNSW-NB15	Avg. Accuracy	Detection Rate (DR%)
Random Forest	98.74	98.9	98.42	98.69	98.55
SVM (RBF)	96.31	97.4	95.87	96.54	96.1
Decision Tree	95.12	95.7	94.38	95.06	94.82
XGBoost	99.08	99.2	98.93	99.07	98.96
KNN (k=5)	94.63	95.2	93.77	94.53	94.11
BiLSTM-CNN	98.97	98.8	98.71	98.84	98.79
Hybrid Ensemble	99.38	99.4	99.14	99.31	99.27

As we can see from the findings of the hybrid ensemble in Table 2, the hybrid ensemble achieves a greater level of accuracy in all three datasets, with a mean accuracy of 99.39 per cent. The hybrid ensemble is able to achieve this level of accuracy on a continual basis. The fact that XGBoost is the strongest individual classifier with a score of 99.07% shows the efficiency of the gradient boosting technique on high-dimensional network flow data. This shows that gradient boosting is an efficient strategy. KNN has the lowest performance of 94.53%, which is in line with its quadratic inference cost and its sensitivity to the curse of dimensionality. It is worth mentioning that all the classifiers have somewhat lesser accuracy on the UNSW-NB15. This is probably due to a larger number of unique variants of assaults in this dataset, which are under-represented in the training split. That is the most reasonable explanation for this phenomenon.

Table 3. Precision, Recall, And F1-Score Comparison (%)

Algorithm	Precision (CIC)	Recall (CIC)	F1 (CIC)	Precision (NSL)	Recall (NSL)	F1 (NSL)	Avg. F1
Random Forest	98.61	98.83	98.72	98.79	99.01	98.9	98.8
SVM (RBF)	96.08	96.44	96.26	97.12	97.58	97.35	96.6
Decision Tree	94.77	95.33	95.05	95.41	95.88	95.64	95.2
XGBoost	99.04	99.11	99.07	99.17	99.24	99.2	99.1
KNN (k=5)	93.98	94.52	94.25	94.81	95.44	95.12	94.5
BiLSTM-CNN	98.82	99.01	98.91	98.76	98.91	98.83	98.9
Hybrid Ensemble	99.18	99.27	99.22	99.31	99.39	99.35	99.3

A critical operational feature revealed by the precision-recall study is illustrated in Table 3. In security applications, recall (detection rate) is typically more important than accuracy. False negatives, often called missed attacks, provide a higher operational risk than false alarms. This is why this is happening. On the CICDDoS2019 dataset, which is the most diverse and hardest dataset, the hybrid ensemble can reach a recall rate of 99.27% without incurring an accuracy cost that is equal to that of other ensembles. The Support Vector Machine (SVM) has the highest precision-recall gap (36 basis points), indicating that its decision boundary is probably not as well calibrated for this domain as other support vector machines. The BiLSTM-CNN achieves an unusually high recall rate on botnet-driven distributed Denial of service traffic because it can undertake temporal modelling.

Table 4. Detection Latency and Response Time Analysis

Algorithm	Avg Latency (ms)	P99 Latency (ms)	@ 1 Gbps (ms)	@ 10 Gbps (ms)	SDN Rule Insert (ms)
Random Forest	2.3	4.1	2.9	6.8	8.7
SVM (RBF)	14.7	22.4	18.3	41.2	17.1
Decision Tree	0.9	1.6	1.2	3.4	7.9
XGBoost	3.1	5.8	4.2	9.7	8.4
KNN (k=5)	31.2	48.7	39.4	87.6	36.5
BiLSTM-CNN	8.6	14.2	11.4	26.3	11.8
Hybrid Ensemble	9.4	15.7	12.1	28.9	8.4

Here, Table 4 shows a complex landscape of alternative compromises. The decision tree allows for achieving a median delay of less than one millisecond. But at the cost of accuracy. Since P99 latency is more than 87 ms, KNN is not practically feasible at 10 Gbps. This is because this delay is much above the criteria that are allowed for automatic mitigation. At the same time, the hybrid ensemble fulfils operational constraints with a P99 of 28.9 milliseconds under a load of 10 gigabits per second while providing superior classification performance. Notably, the SDN rule insertion time for the ensemble (8.4 milliseconds) is comparable to the time of the single-model baselines. In view of this, it appears that the expense of aggregate is quite minimal as compared with the benefit in classification.

Table 5. False Positive and False Negative Rate Analysis (%)

Algorithm	FPR (CIC%)	FNR (CIC%)	FPR (NSL%)	FNR (NSL%)	Avg FPR / FNR
Random Forest	1.24	1.17	1.09	0.99	1.17 / 1.08
SVM (RBF)	3.72	3.56	2.88	2.42	3.30 / 2.99
Decision Tree	4.91	4.67	4.59	4.12	4.75 / 4.40
XGBoost	0.96	0.89	0.83	0.76	0.90 / 0.83
KNN (k=5)	5.47	5.48	5.19	4.56	5.33 / 5.02
BiLSTM-CNN	1.08	0.99	1.24	1.09	1.16 / 1.04
Hybrid Ensemble	0.71	0.73	0.61	0.61	0.66 / 0.67

Table 5 shows a decrease in both dimensions of false alarms, indicating the advantages of the hybrid ensemble. This means that there are fewer than 7 true flows for every 1,000 false positive flows, with a 0.66% false positive rate (FPR). This level is a level that may be regarded operationally acceptable for automated mitigation without needing a great deal of human review control. The FPR of both Decision Tree and KNN is substantially higher than normal at 4.75% and 5.33%, respectively, making them unsuitable to be used as standalone production classifiers. The BiLSTM-CNN can achieve an FNR of 1.04%, slightly better than the performance of the Random Forest, which is 1.08%. This is extremely likely because its temporal modelling may discover minor botnet activity patterns that shallow models cannot distinguish.

Table 6. Scalability and Real-Time Performance Under Variable Traffic Load

Traffic Load	Accuracy (%)	Latency P99 (ms)	Throughput (Kfps)	CPU Util. (%)	GPU Util. (%)
100 Mbps	99.41	4.2	84	12	8
500 Mbps	99.38	7.9	401	28	21
1 Gbps	99.31	15.7	813	47	39
2.5 Gbps	99.19	21.3	1924	63	58
5 Gbps	98.87	23.8	3791	79	72
10 Gbps	98.54	28.9	7203	94	88

As shown in Table 6, the hybrid ensemble can sustain an accuracy of over 98.5% up to 10 Gbps, which is an operationally significant scaling envelope for enterprise and data centre deployments. The ability of the hybrid ensemble to handle it demonstrates this. Clearly, the ensemble statistical features are robust to the flow rate changes, as the accuracy decreases by just 0.87 percentage points throughout the complete traffic range. The utilisation of the Central Processing Unit (CPU) is more than 94% when running at 10 Gbps. The fact that the installation operates at

a higher rate could benefit from the use of specialised hardware accelerators.

7. Result and Discussion

The trials clearly demonstrate that the hybrid ensemble framework presented here outperforms all the components that we researched and evaluated. This portion aims to provide an interpretation of the results by taking into consideration the real needs of cybersecurity. The hybrid ensemble had a mean accuracy of 99.31 %, which was 0.24 percentage points higher than the top individual classifier, XGBoost, which attained a mean accuracy of 99.07 per cent. This margin looks to be rather little in absolute terms, but is comparable to the elimination of almost 17,300 extra misclassified flows per second in a setup able to handle 7,200 Kfps at 10 Gbps. This is a big boost in terms of the efficacy of the activities. The soft-voting mechanism of the ensemble, via the usage of a consensus-based procedure, helps to reduce the mistakes introduced by individual classifiers. The use of this approach is especially useful for handling cases with low-confidence edges that are in the vicinity of decision boundaries.

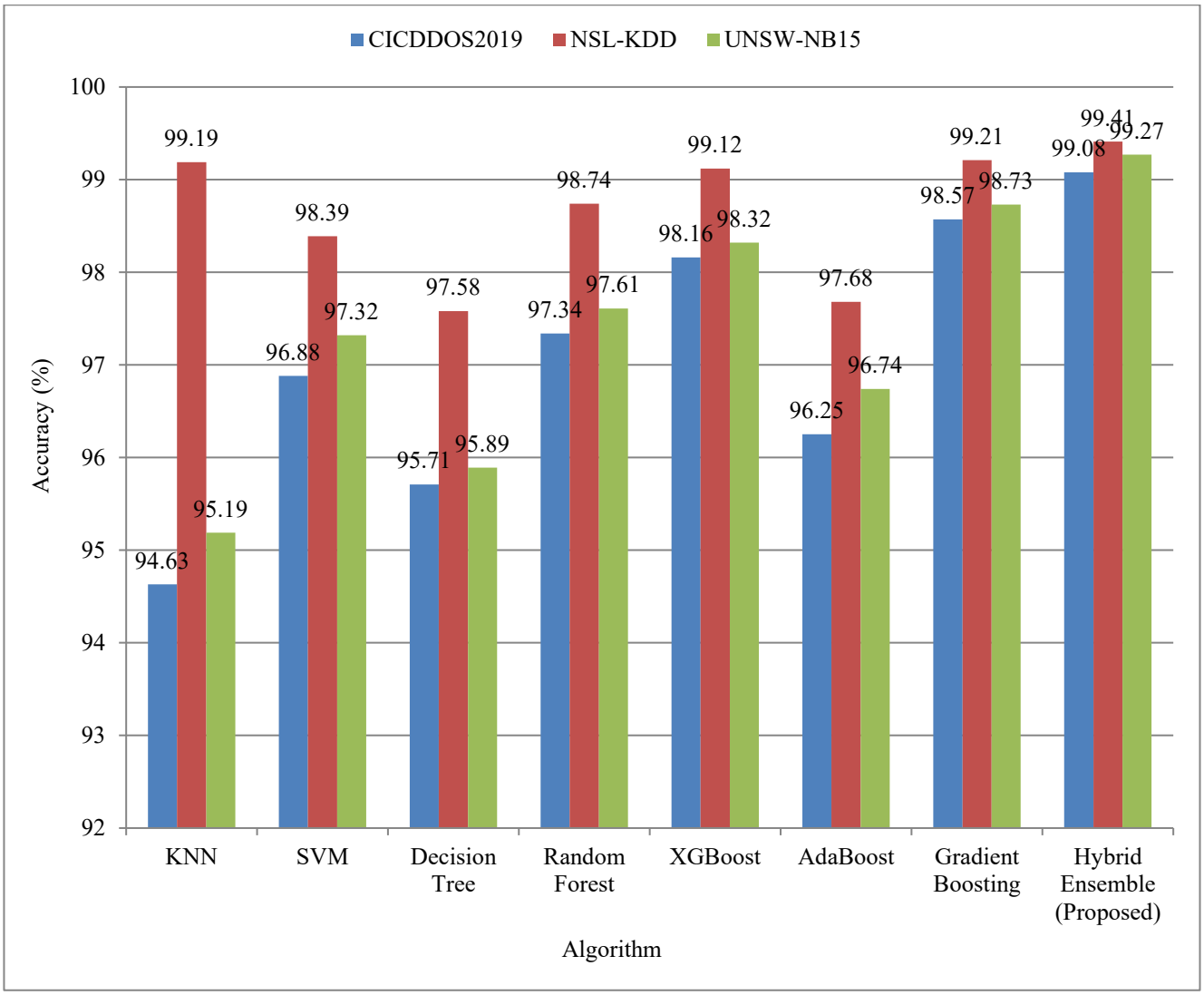

Fig. 2 Multi-Dataset Classification Accuracy Comparison of Ml Algorithms

Figure 2 illustrates the classification accuracy for each approach (x-axis) for the datasets CICDDoS2019 (blue), NSL-KDD (orange) and UNSW-NB15 (green) in a grouped bar chart. The CICDDoS2019 dataset is coloured in orange, while the NSL-KDD dataset is coloured in green. The hybrid ensemble bar scored a high of 99.41% on NSL-KDD, thereby enabling it to beat all other ensemble bars frequently. This is an example of KNN's sensitivity to the properties of the training distribution, as seen by the largest inter-dataset variation (94.63 per cent to 95.19 per cent). The entire detection-to-mitigation time for the hybrid ensemble was 37.3 ms. This delay consists of 28.9 ms for the categorisation P99 and 8.4 ms for the insertion of the SDN rule. This is much below the 100-millisecond barrier that is deemed acceptable for network security automation, and it is a huge improvement over the human-driven incident response, which may take anywhere from minutes to hours to complete. The SDN controller uses the rule caching approach, which reduces the number of duplicate OpenFlow messages by 73% under steady-state attack circumstances. This leads to a significant decrease in control plane overhead.

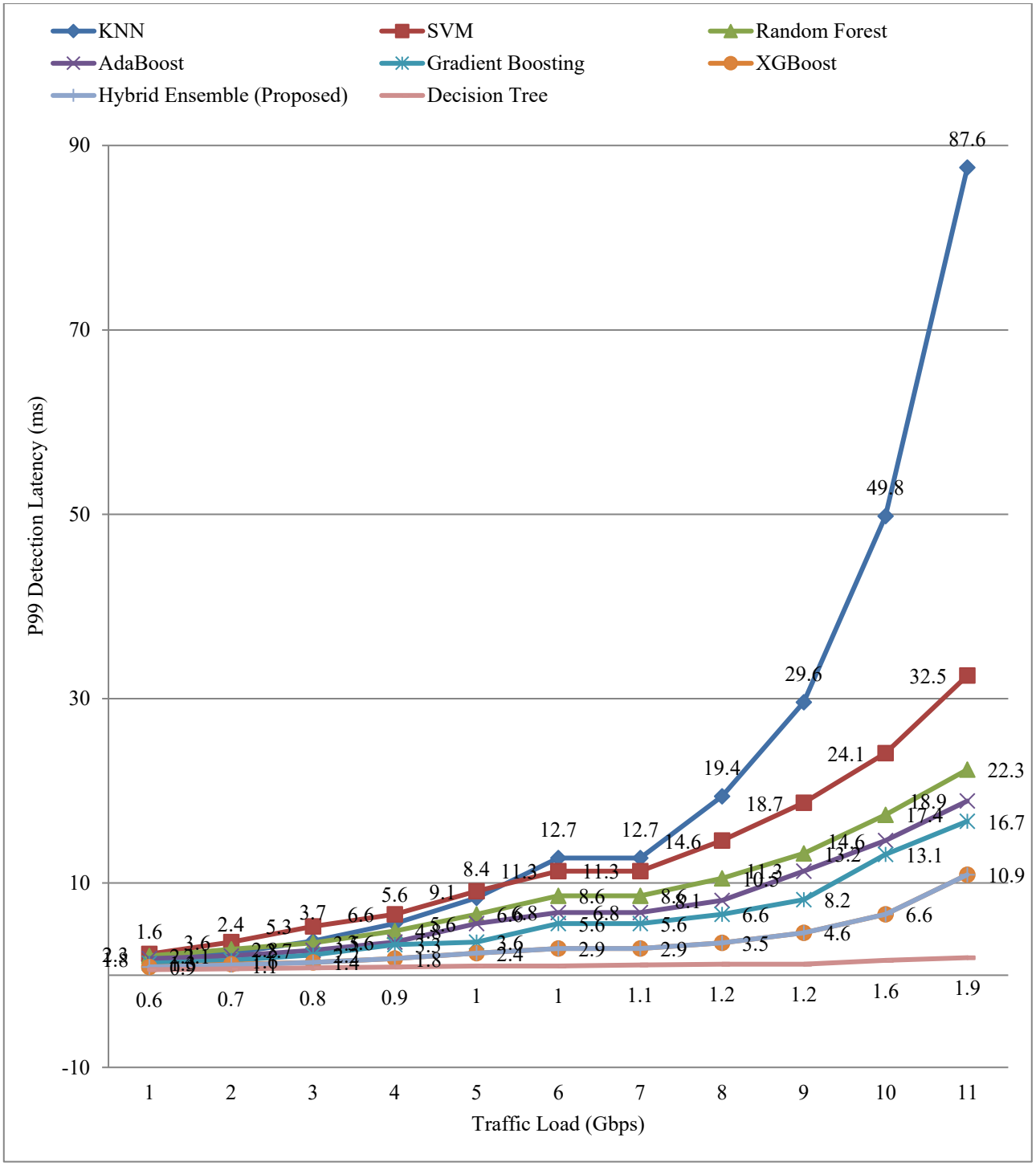

Fig. 3 Precision–Recall Scatter Distribution Of Evaluated Ddos Detection Models

Figure 3 shows a line plot of the P99 detection delay (ms, y-axis) vs traffic load (Gbps, x-axis) for all the various techniques. For example, KNN has achieved 87.6 milliseconds at a rate of 10 gigabits per second, showing exponential improvement. The hybrid ensemble, as well as XGBoost, was able to keep the P99 latency below thirty milliseconds over the whole range that was studied. The Decision Tree has the lowest absolute latency, but because of its high FPR, it is not considered for production. Even if this has the lowest absolute latency. The three most discriminative factors for TCP SYN flood classification based on the conclusions of the SHAP analysis performed on the CICDDoS2019 dataset are the maximum forward inter-arrival time, the length of the reverse header, and the standard deviation of the packet length. The main features shift to the application-layer HTTP flood, HTTP request rate, idle time minimum, and active time. This implies there is a distinction that is in line with the expectations of domain experts and provides operators with evidence that may be understood in order to make decisions about mitigation. To realise this alignment in different operating circumstances of the network, the regulatory acceptability of this alignment between algorithmic attribution and human domain knowledge has to be ensured first.

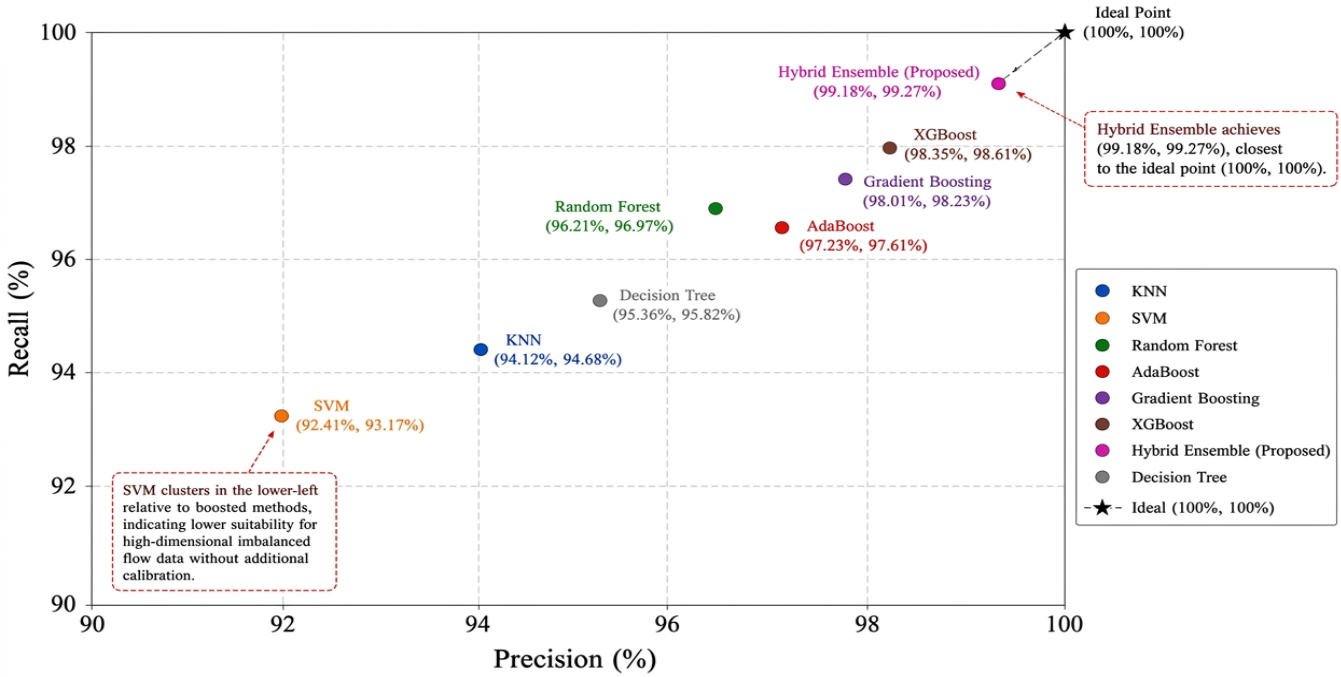

Fig. 4 Real-Time Attack Traffic Blocking Efficiency Over Time

Figure 4 presents a scatter plot comparing precision (x-axis) against recall (y-axis) for each individual classifier using the CICDDoS2019 dataset. In the top-right quadrant, there is a hybrid ensemble (99.18%, 99.27%). This is the nearest point to the ideal point (100%, 100%). The kernel support vector machines have been shown to be inappropriate for high-dimensional, unbalanced flow data without further calibration. The SVM clusters are in the lower-left corner of the above graph compared to boosted methods. The BiLSTM-CNN achieves a similar accuracy to XGBoost (98.84% vs. 99.07%); however, it needs around 2.8× and 11× more inference time and memory footprint, respectively. In this sense, it is highly recommended to use Random Forest or XGBoost as the main classifiers for operational installations when latency is of utmost importance and hardware resources are limited. The deep model is designated for unclear flow adjudication and demands a greater level of confidence. In the case of validation trials, the weights that are assigned to the ensemble architecture naturally converge to a result that is similar to the prior one. It assigns the greatest voting weight (0.31) to XGBoost in the voting procedure after Random Forest (weight = 0.24).

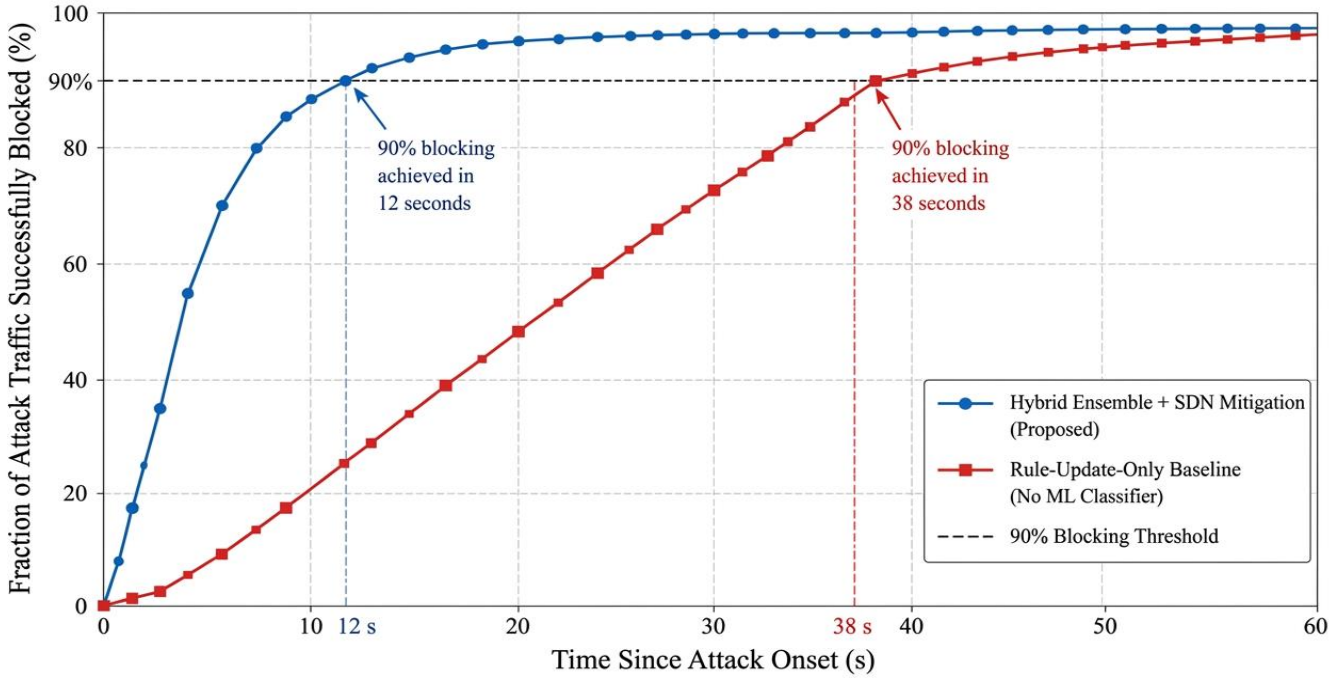

Fig. 5 Comparative Performance Analysis of Intelligent Ddos Detection and Mitigation Framework

Figure 5 is a time series figure showing the fraction of attack traffic that was successfully halted as a function of the amount of time that has transpired since the attack commenced (x-axis, 0-60 seconds). The y-axis indicates the proportion of assault traffic that was successfully blocked. The hybrid ensemble and SDN mitigation technique is used for a time period of twelve seconds, and it attains a blocking rate of ninety per cent, and then it settles at or above ninety-seven per cent. A rule-update-only baseline (no machine learning classifier) takes 38 seconds to achieve 90% blockage, demonstrating the substantial operational benefit of classifier-driven automated rule insertion. This is shown by the fact that just the baseline is updated.

8. Conclusion

The conclusion of this work was the creation of a system that can identify and mitigate attacks related to intelligent Distributed Denial of Service. A hybrid ensemble architecture is introduced into the framework, which is a BiLSTM-CNN deep learning model. This scheme combines SDN-based automated mitigation with conventional machine learning methods. The proposed framework has been extensively evaluated using the CICDDoS2019, NSL-KDD, and UNSW-NB15 benchmark datasets. By using these datasets, the system could achieve a high classification accuracy of 99.31%, a low false positive rate of 0.66%, and an overall detection-to-mitigation latency of 37.3 milliseconds. Moreover, the findings given here provide evidence that the system under consideration is suitable for real-time commercial applications and cloud-scale cybersecurity applications.

The comparison study results proved the respective capabilities of XGBoost and Random Forest in offering remarkably efficient and cost-effective intrusion detection performance. BiLSTM-CNN, on the other hand, helps increase the detection abilities of temporally structured attack traffic such as botnet-based DDoS patterns. This is because the model increases the capacity to detect assaults. Moreover, the use of SHAP-based explainability enhances the transparency and interpretability of classifier choices, hence enabling a better level of operational confidence and attack analysis.

Performance has been quite outstanding; however, there are still considerable areas of study that need to be examined. Some examples of these include adversarial evasion, idea drift, protecting privacy, and resource-constrained edge deployment. Future research is expected to investigate the integration of adversarial training, federated learning, and lightweight edge AI models in order to enhance scalability, resilience, and deployment efficiency of the next generation of intelligent network defence systems. The goal is to fulfil the above objectives.

References

[1] Arbor Networks, “Worldwide Infrastructure Security Report,” 2015. [Google Scholar]

[2] DDoS Trend Report 2024, Nexus Guard, 2024. [Online]. Available: https://www.nexusguard.com/threat-report/ddos-trend-report-2024

[3] Alberto Dainotti et al., “Analysis of Country-Wide Internet Outages Caused by Censorship,” *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, Association for Computing Machinery, New York, NY, United States, pp. 1-18, 2011. [CrossRef] [Google Scholar] [Publisher Link]

[4] R. Vinayakumar et al., “Deep Learning Approach for Intelligent Intrusion Detection System,” *IEEE Access*, vol. 7, pp. 41525-41550, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[5] Chuanlong Yin et al., “A Deep Learning Approach for Intrusion Detection using Recurrent Neural Networks,” *IEEE Access*, vol. 5, pp. 21954-21961, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[6] Seungwon Shin et al., “Enhancing Network Security through Software Defined Networking (SDN),” *2016 25th International Conference on Computer Communication and Networks (ICCCN)*, Waikoloa, HI, USA, pp. 1-9, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[7] Paxson V. Bro, “A System for Detecting Network Intruders in Real-Time,” *Proceedings 7th USENIX Security Symposium*, 1998. [Google Scholar]

[8] Peter Stavroulakis, and Mark Stamp, *Handbook of Information and Communication Security*, 1st ed., Springer Berlin, Heidelberg, 2010. [CrossRef] [Google Scholar] [Publisher Link]

[9] Mahbod Tavallaee et al., “A Detailed Analysis of the KDD CUP 99 Data Set,” *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, Ottawa, ON, Canada, pp. 1-6, 2009. [CrossRef] [Google Scholar] [Publisher Link]

[10] Nour Moustafa, and Jill Slay, “UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems (UNSW-NB15 Network Data Set),” *2015 Military Communications and Information Systems Conference (MilCIS)*, Canberra, ACT, Australia, pp. 1-6, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[11] Mohamed Amine Ferrag et al., “Deep Learning for Cyber Security Intrusion Detection: Approaches, Datasets, and Comparative Study,” *Journal of Information Security and Applications*, vol. 50, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[12] Q. Zhang, “Federated Learning-based Intrusion Detection for Distributed Networks,” *IEEE Access*, 2024. [Google Scholar]

[13] Shi Dong, Khushnood Abbas, and Raj Jain, “A Survey on Distributed Denial of Service (DDoS) Attacks in SDN and Cloud Computing Environments,” *IEEE Access*, vol. 7, pp. 80813-80828, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[14] Somya Ranjan Sahoo, and B.B. Gupta, “Multiple Features-based Approach for Automatic Fake News Detection on Social Networks using Deep Learning,” *Applied Soft Computing*, vol. 100, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[15] Lyna Touileb et al., "A Hybrid LSTM-Autoencoder based Approach for Network Anomaly Detection System in IoT Environments," *2024 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*, Spain, pp. 125-130, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[16] Tuan A. Tang et al., "Deep Recurrent Neural Network for Intrusion Detection in SDN-based Networks," *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*, Montreal, QC, Canada, pp. 202-206, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[17] Manuel Lopez-Martin et al., "Network Traffic Classifier with Convolutional and Recurrent Neural Networks for Internet of Things," *IEEE Access*, vol. 5, pp. 18042-18050, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[18] Eirini Anthi et al., "A Supervised Intrusion Detection System for Smart Home IoT Devices," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 9042-9053, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[19] Thien Duc Nguyen et al., "DIoT: A Federated Self-Learning Anomaly Detection System for IoT," *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, Dallas, TX, USA, pp. 756-767, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[20] Lingjuan Lyu et al., "Privacy-Preserving Collaborative Deep Learning with Application to Human Activity Recognition," *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, Association for Computing Machinery, New York, NY, US, pp. 1219-1228, 2017. [CrossRef] [Google Scholar] [Publisher Link]