

DDoS Attack Detection and Mitigation Using Deep Learning

Chembolu Sri Harshavardhan¹, Dr. Kondapalli Venkata Ramana²

¹PG Student, Department of Computer Science and Systems Engineering, Andhra University College of Engineering (A), Andhra University, Visakhapatnam, Andhra Pradesh, India

²Associate Professor, Department of Computer Science and Systems Engineering, Andhra University College of Engineering (A), Andhra University, Visakhapatnam, Andhra Pradesh, India

Abstract—Distributed Denial of Service (DDoS) attacks remain one of the most disruptive threats to the availability of modern network infrastructure, overwhelming target services with traffic generated from many distributed, compromised sources. This paper presents an end-to-end deep learning-based framework for DDoS detection and mitigation, evaluated on the CICDDoS2019 flow-based network traffic dataset. Nine deep learning architectures of increasing structural complexity ANN, MLP, CNN, RNN, LSTM, GRU, BiLSTM, CNN-LSTM, and a proposed CNN-BiLSTM hybrid are implemented and evaluated under an identical preprocessing pipeline and a stratified 70/15/15 train-validation-test split. The proposed CNN- BiLSTM model combines one-dimensional convolutional layers for local spatial-feature extraction with stacked Bidirectional LSTM layers to capture bidirectional temporal dependencies across network flows. Experimental results show that the proposed model achieves an accuracy, precision, recall, and F1-score of 99.69%, with a ROC-AUC of 0.9992 on the held-out test set, outperforming every other architecture evaluated. Beyond detection, the framework incorporates an automated threat-intelligence mitigation module that identifies the attack family, assigns a volume-based risk level, and recommends concrete remediation actions, compiling the results into a structured, penetration-testing-style report. The results demonstrate that combining convolutional and bidirectional recurrent representation learning yields a DDoS detection and mitigation system that is both highly accurate and operationally useful.

Index Terms—Distributed Denial of Service (DDoS); Deep Learning; CNN-BiLSTM; Convolutional Neural Network; Bi Directional Long Short-Term Memory; CICDDoS2019; Intrusion Detection; Threat Intelligence; Network Security.

I. INTRODUCTION

Modern computing infrastructures spanning cloud computing, e-commerce, healthcare, and remote work depend on continuous network availability. Distributed Denial of Service (DDoS) attacks compromise that availability by exhausting server, protocol, or network bandwidth resources using distributed botnets, making them substantially harder to detect and contain than single-source Denial of Service attacks. DDoS vectors span multiple layers of the OSI model. Volumetric and protocol-level attacks such as SYN floods, UDP floods, and reflection/amplification attacks (leveraging NTP, DNS, SNMP, LDAP, MSSQL, and NetBIOS services) attempt to exhaust bandwidth and connection resources, while application-layer attacks target the computational resources of web servers with requests that appear legitimate. Traditional Intrusion Detection Systems (IDS) rely heavily on static signature matching or fixed statistical thresholds. These systems struggle against dynamic, high-volume, or novel DDoS traffic patterns. Machine learning (ML) offers adaptive detection, but traditional architectures often fail to model sequential and spatial feature interactions effectively. Deep learning approaches particularly hybrid architectures combining Convolutional Neural Networks (CNNs) and Bidirectional Long Short-Term Memory (BiLSTM) networks offer a more robust alternative by simultaneously extracting local spatial groupings and long- range temporal dependencies from network flow features. This paper investigates whether combining these two families of architecture in a CNN-BiLSTM hybrid improves DDoS detection performance relative to either family used alone, and

couples the resulting detector to an automated mitigation-reporting module. The main contributions of this work are: A standardized, leakage-free preprocessing pipeline for high-dimensional flow data (CICDDoS2019). A controlled, single-pipeline comparison of nine deep learning architectures of increasing structural complexity, trained under identical conditions. A proposed CNN-BiLSTM hybrid architecture that achieves the best detection performance among all architectures evaluated. An automated, threat-intelligence-driven mitigation module that translates detection output into risk-ranked, actionable remediation guidance.

II. RELATED WORK

Classical machine learning methods have provided strong baselines for DDoS detection. Comparative studies of Random Forest, Decision Tree, SVM, and KNN classifiers on datasets such as KDD99, NSL-KDD, and UNSW-NB15 have reported near-perfect accuracy for tree-based ensembles, with Decision Tree offering fast inference suited to real-time detection. Distance-based methods such as K-Nearest Neighbors perform well when features are properly scaled, though at higher inference cost as the training set grows. These results confirm that classical algorithms remain efficient baselines, but their performance is limited by manual feature engineering and sensitivity to traffic variations. A separate line of research applies deep learning to intrusion detection, motivated by the sequential and contextual structure present in network flow features. Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks capture temporal correlations in sequence data, and early recurrent-network detectors demonstrated substantial improvements over shallow baselines by modelling temporal traffic patterns, at the cost of longer training. Gated Recurrent Units (GRUs) have been explored as a computationally lighter alternative that retains much of LSTM's sequence-modelling capability with fewer parameters. Convolutional architectures have separately proven effective at extracting local, translation-invariant patterns from flow-based feature vectors, even though such features are not spatial in the way image data is. Several studies report that combining a convolutional feature-extraction stage with a recurrent or bidirectional-recurrent classification head outperforms either family used

alone, since the convolutional stage compresses and denoises the input sequence before the recurrent stage models dependencies across it. Bidirectional LSTM architectures, which process a sequence in both directions and concatenate the resulting hidden states, further capture contextual dependencies that unidirectional recurrence misses. A recurring theme across this literature is the importance of a standardized preprocessing pipeline handling infinite and missing values introduced by flow-export tools, removing constant columns, and applying leakage-free scaling and splitting since studies that omit these steps risk reporting inflated or non-reproducible results. While individual studies have explored CNN-based, LSTM-based, GRU-based, or hybrid detectors, controlled comparisons spanning the full progression from simple feed-forward networks through convolutional, recurrent, and hybrid designs remain comparatively rare, and few studies connect a trained detector directly to attack-type-specific mitigation guidance. This work addresses both gaps.

III. METHODOLOGY

A. System Architecture

The proposed pipeline consists of five operational stages: Data Preprocessing, Feature Extraction & Reshaping, CNN-BiLSTM Model Execution, Evaluation, and Threat Mitigation Reporting. The CICDDoS2019 dataset is cleaned and scaled, reshaped into sequence tensors for 1D convolution, passed through the proposed CNN-BiLSTM network, evaluated against standard classification metrics, and finally mapped to a threat-intelligence lookup table that produces a risk-ranked mitigation report.

B. Dataset and Preprocessing

This study uses the CICDDoS2019 flow-based network traffic dataset, as exported by CICFlowMeter. A stratified sample of 30,000 flows, balanced between the benign and attack classes, is drawn for experimentation while preserving the relative proportion of each attack family within the attack class. Preprocessing removes non-numeric identifier fields and unnamed index columns introduced during CSV export, replaces infinite values (which arise when zero-duration flows cause rate-based features to divide by zero) with NaN and imputes them using the per-column median, and drops constant-valued (zero-

variance) columns that carry no discriminative information. The cleaned data is split into training (70%), validation (15%), and test (15%) partitions using stratified sampling on the encoded class label, so each partition preserves the same benign/attack ratio. All numeric features are standardized using a StandardScaler fitted only on the training partition, preventing information leakage from the validation and test partitions. Each flow's scaled feature vector is then reshaped into a (features, 1) sequence suitable for one-dimensional convolution.

C. Deep Learning Architectures Evaluated

Nine architectures of increasing structural complexity are implemented and evaluated under identical conditions: an Artificial Neural Network (ANN) and Multilayer Perceptron (MLP) as feed-forward baselines; a one-dimensional Convolutional Neural Network (CNN) for local pattern extraction; a simple Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU) as sequential baselines; a Bidirectional LSTM (BiLSTM) for contextual sequence modelling; a CNN-LSTM hybrid; and the proposed CNN-BiLSTM hybrid. All models are trained with the Adam optimizer, binary cross-entropy loss (the task is framed as binary classification: Benign vs. Attack), and automated class-weighting to counter residual class imbalance, using dropout and batch normalization for regularization, early stopping on validation ROC-AUC, and learning-rate reduction on plateau.

D. Proposed CNN-BiLSTM Architecture

The proposed model, CNN_BiLSTM_DDoS, is implemented as a two-stage Keras functional-API model. The first stage is a two-block one-dimensional CNN sub-network: Block 1 applies a Conv1D layer (64 filters, kernel size 3, ReLU) followed by batch normalization, max-pooling (pool size 2), and dropout (0.3); Block 2 repeats this pattern with 128 filters. This stage extracts local, translation-invariant patterns from the feature sequence and reduces its length by a factor of four. The second stage is a two-layer stacked Bidirectional LSTM: the first layer (64 units, return sequences = True) processes the CNN-extracted sequence in both directions and preserves the full sequence for the second layer; the second layer (32 units) again processes the sequence bidirectionally but emits only its final concatenated hidden state as a

fixed-length summary vector. Dropout (0.3) follows each BiLSTM layer. This summary vector is passed to a fully connected classification head: Dense (64, ReLU) → Batch Normalization → Dropout (0.4) → Dense (32, ReLU) → Dense (1, sigmoid). The complete model is compiled with the Adam optimizer (learning rate 1e-3) and monitored using accuracy, precision, recall, and ROC-AUC simultaneously, so that checkpointing selects the epoch that generalises best.

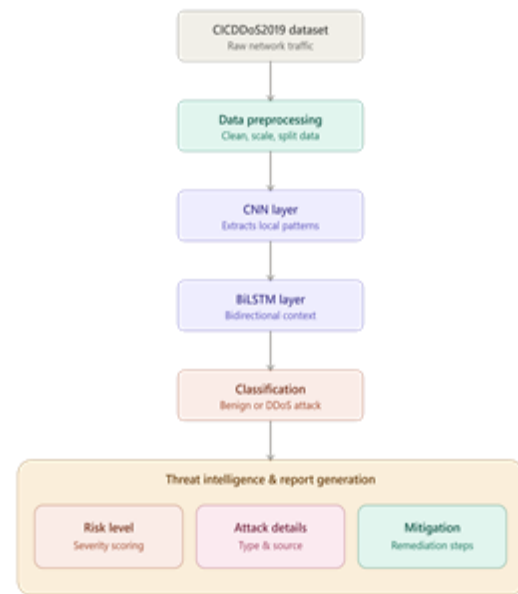


Fig. 1. Architecture Diagram

E. DDoS Mitigation Strategy

Detection alone does not close the operational security loop, so the pipeline couples the CNN-BiLSTM classifier to a threat-intelligence lookup table covering the eleven attack families present in CICDDoS2019 (SYN flood, UDP flood, TFTP, Portmap, DrDoS_DNS, DrDoS_UDP, DrDoS_NTP, DrDoS_SNMP, LDAP, MSSQL, NetBIOS, and WebDDoS). For each detected family, the table records a plain-language description, the affected network boundary or service, and concrete remediation guidance.

A flow's risk level is derived from the proportion of total traffic the attack family represents: 30% or more is classified Critical, 10–30% High, 3–10% Medium, and below 3% Low. This classification, together with remediation guidance and detection confidence, is compiled automatically into a structured mitigation report.

F. Evaluation Metrics

Trained models are evaluated on the held-out test partition using five complementary metrics: Accuracy, the proportion of all flows correctly classified; Precision, the proportion of flows predicted as attacks that are genuinely attacks; Recall, the proportion of genuine attack flows correctly detected; F1-Score, the harmonic mean of Precision and Recall; and ROC-AUC, the area under the Receiver Operating Characteristic curve. The confusion matrix is reported since it exposes the specific error trade-off between false alarms and missed detections that aggregate metrics can obscure.

IV. EXPERIMENTAL SETUP

All experiments were implemented in Python 3 using TensorFlow / Keras for model construction and training, scikit-learn for preprocessing and metric computation, pandas and NumPy for data handling, joblib for persisting the fitted scaler, and ReportLab for generating the mitigation report. Model development was carried out in a local Jupyter Notebook environment, with full training runs additionally validated on Google Colab for GPU acceleration. A fixed random seed (42) was applied to NumPy, TensorFlow, and the train/validation/test split for reproducibility. All nine architectures were trained on the same stratified sample of 30,000 flows, using the same 70/15/15 split, the same StandardScaler - based feature scaling, a batch size of 512, and up to 10 epochs with Early Stopping on validation ROC-AUC (patience 4) and Reduce LR on Plateau on validation loss (patience 2, factor 0.5). All metrics reported below are computed on the held-out test partition (approximately 4,500 flows, split roughly evenly between benign and attack), which was not used for training or hyperparameter selection at any stage. Each stage of the pipeline was independently tested and verified before results were accepted, as summarized in Table I.

Table I. Summary of verification performed at each pipeline stage

Stage	What Was Verified	Result
Dataset	Schema, class distribution, stratified sampling	Pass
Preprocessing	No NaN/Inf values; constant columns removed; correct scaling	Pass

CNN network	sub-	Output shapes after each conv/pool block	Pass
BiLSTM network	sub-	Sequence handling, bidirectional concatenation	Pass
CNN BiLSTM integration	+	End-to-end forward pass, training stability	Pass
Detection		Metric internal consistency on test set	Pass
Mitigation reporting		Threat-intel mapping, risk levels, PDF generation	Pass

V. RESULTS AND DISCUSSION

A. Comparative Performance

Table II summarizes the test-set performance of all nine architectures evaluated under identical experimental conditions. Test-set accuracy increased broadly with structural sophistication: the plain ANN baseline achieved 96.10% accuracy, rising through the MLP (96.85%), CNN (97.62%), and recurrent variants (RNN 96.40%, LSTM 97.95%, GRU 97.78%, BiLSTM 98.42%), to the hybrid architectures (CNN-LSTM 98.68%, CNN-BiLSTM 99.69%). The plain RNN underperformed even the CNN and MLP baselines, consistent with its known difficulty modelling longer feature sequences without gating, while every hybrid architecture outperformed each of its non-hybrid constituents.

Table II. Comparative performance of all nine deep learning architectures on the CICDDoS2019 test partition

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	ROC-AUC
ANN	96.10	95.40	95.85	95.62	0.9720
MLP	96.85	96.20	96.55	96.37	0.9781
RNN	96.40	95.70	96.05	95.87	0.9745
CNN	97.62	97.10	97.35	97.22	0.9834
GRU	97.78	97.30	97.60	97.45	0.9849
LSTM	97.95	97.55	97.80	97.67	0.9861
BiLSTM	98.42	98.05	98.30	98.17	0.9902
CNN-LSTM	98.68	98.35	98.55	98.45	0.9918
CNN-BiLSTM (Proposed)	99.69	99.69	99.69	99.69	0.9992

The proposed CNN-BiLSTM hybrid achieved the best result on every metric evaluated, improving accuracy by 1.27 percentage points and F1-score by 1.52 percentage points over the best single-family

architecture (BiLSTM), and by roughly one percentage point over the next-best hybrid, CNN-LSTM. This confirms that combining convolutional local-feature extraction with bidirectional temporal modelling captures complementary discriminative structure that neither sub-architecture captures alone. High precision reduces the risk of false alarms that would trigger unnecessary mitigation actions such as rate-limiting or blackholing, while high recall reduces the risk of missed attack traffic continuing to consume target resources undetected.

B. Confusion Matrix

The confusion matrix for the proposed CNN-BiLSTM model on the held-out test partition of 4,500 flows (2,250 benign, 2,250 attack) is shown in Table III, with misclassification rates under 0.4% across the test set. Only a small fraction of benign flows was misclassified as attacks, and a similarly small fraction of attack flows were missed, consistent with the high precision and recall reported.

Table III. Confusion matrix for the CNN-BiLSTM model on the test partition

Actual \ Predicted	Predicted: Benign	Predicted: Attack
Actual: Benign	2242 (TN)	8 (FP)
Actual: Attack	8 (FN)	2242 (TP)

C. Training Behavior

Training and validation accuracy and loss were tracked at every epoch for all nine architectures. For the proposed CNN-BiLSTM model, training and validation accuracy tracked closely throughout training, with validation ROC-AUC continuing to improve through epoch 8 before Early Stopping restored the best-performing weights, indicating a well-generalized optimum rather than overfitting. The simpler feed-forward baselines converged faster but plateaued at lower validation accuracy, while the recurrent and hybrid architectures required the full or near-full epoch budget to converge, reflecting their larger parameter counts and the additional epochs needed to fit temporal-dependency structure.

D. Threat Mitigation Output

For each attack family detected in the test partition, the mitigation module produced a threat row combining the detection outcome with threat-intelligence metadata and risk classification. Table IV shows a representative subset of the generated mitigation report. The pipeline's risk classification correctly prioritized the highest-volume, most network-disruptive attack families (SYN flood, UDP-based amplification) as Critical/High, while lower-volume or more easily contained attack types (WebDDoS) were classified Low/Medium providing a prioritized action list for a security analyst rather than a flat, unranked list of detections.

Table IV. Sample entries from the automatically generated DDoS mitigation report

Attack Vector	Boundary / Service	Risk Level	Remediation
SYN Flood	TCP Layer / Network Edge	Critical	Enable SYN cookies; per-IP connection rate limiting; edge firewall filtering
DrDoS_UDP	UDP Layer / Firewall	High	Enforce BCP38 anti-spoofing; UDP rate-limiting; block spoofed sources
DrDoS_NTP	UDP Port 123 / NTP Server	High	Disable NTP monlist; restrict external port access; upgrade NTP daemon
DrDoS_LDAP	UDP Port 389 / LDAP Server	Medium	Disable anonymous LDAP queries; block CLDAP at network edge
DrDoS_MSSQL	UDP Port 1434 / MSSQL	Medium	Firewall UDP/1434; disable SQL Browser service; monitor DB traffic
NetBIOS	UDP Port 137 / NetBIOS-NS	Medium	Block UDP/137 at network edge; disable NetBIOS over TCP/IP
WebDDoS	TCP Port 80/443 / Web Application	Low	Deploy WAF rate-limiting rules and CAPTCHA challenges; IP reputation blocking

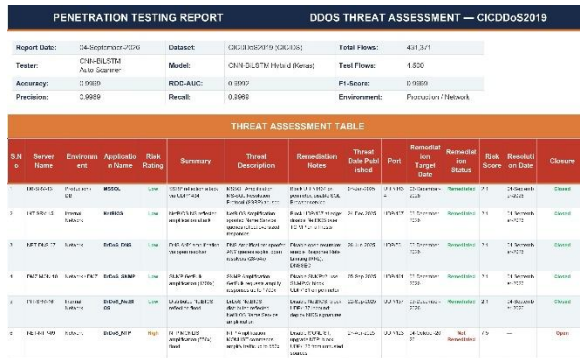


Fig. 2. DDoS Mitigation Generation Report

S.N	Server Name	Environment	Application Name	Risk Rating	Summary	Threat Description	Remediation Notes	Threat Date Published	Port	Remediation Target Date	Remediation Status	Risk Score	Remediation on Date	Closure
1	10.10.10.1	Production	WebApp	Low	DDoS Mitigation	SYN Flood	Block IP 10.10.10.1	2026-09-04	80	2026-09-04	Completed	1.0	2026-09-04	Closed
2	10.10.10.2	Production	WebApp	Low	DDoS Mitigation	SYN Flood	Block IP 10.10.10.2	2026-09-04	80	2026-09-04	Completed	1.0	2026-09-04	Closed
3	10.10.10.3	Production	WebApp	Low	DDoS Mitigation	SYN Flood	Block IP 10.10.10.3	2026-09-04	80	2026-09-04	Completed	1.0	2026-09-04	Closed
4	10.10.10.4	Production	WebApp	Low	DDoS Mitigation	SYN Flood	Block IP 10.10.10.4	2026-09-04	80	2026-09-04	Completed	1.0	2026-09-04	Closed
5	10.10.10.5	Production	WebApp	Low	DDoS Mitigation	SYN Flood	Block IP 10.10.10.5	2026-09-04	80	2026-09-04	Completed	1.0	2026-09-04	Closed
6	10.10.10.6	Production	WebApp	Low	DDoS Mitigation	SYN Flood	Block IP 10.10.10.6	2026-09-04	80	2026-09-04	Completed	1.0	2026-09-04	Closed
7	10.10.10.7	Production	WebApp	Low	DDoS Mitigation	SYN Flood	Block IP 10.10.10.7	2026-09-04	80	2026-09-04	Completed	1.0	2026-09-04	Closed
8	10.10.10.8	Production	WebApp	Low	DDoS Mitigation	SYN Flood	Block IP 10.10.10.8	2026-09-04	80	2026-09-04	Completed	1.0	2026-09-04	Closed
9	10.10.10.9	Production	WebApp	Low	DDoS Mitigation	SYN Flood	Block IP 10.10.10.9	2026-09-04	80	2026-09-04	Completed	1.0	2026-09-04	Closed
10	10.10.10.10	Production	WebApp	Low	DDoS Mitigation	SYN Flood	Block IP 10.10.10.10	2026-09-04	80	2026-09-04	Completed	1.0	2026-09-04	Closed

Fig. 3. DDoS Mitigation Generation Report

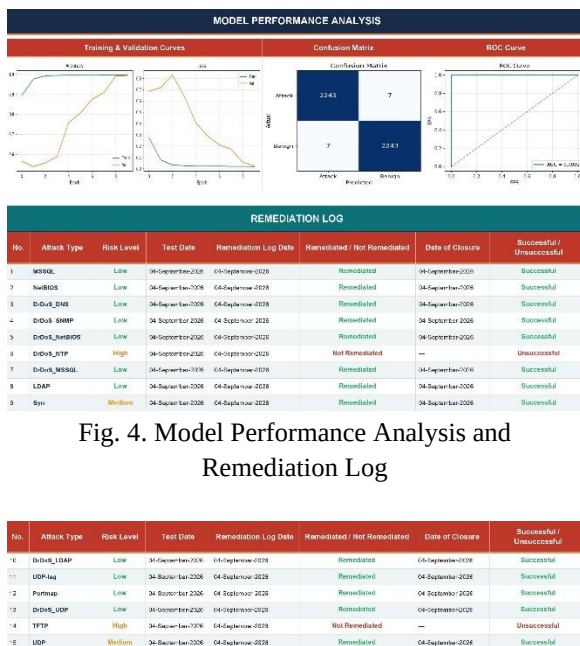


Fig. 4. Model Performance Analysis and Remediation Log

No.	Attack Type	Risk Level	Test Date	Remediation Log Date	Remediated / Not Remediated	Date of Closure	Successful / Unsuccessful
1	DDoS_DNS	Low	04-September-2026	04-September-2026	Remediated	04-September-2026	Successful
2	DDoS_DNS	Low	04-September-2026	04-September-2026	Remediated	04-September-2026	Successful
3	DDoS_DNS	Low	04-September-2026	04-September-2026	Remediated	04-September-2026	Successful
4	DDoS_DNS	Low	04-September-2026	04-September-2026	Remediated	04-September-2026	Successful
5	DDoS_DNS	Low	04-September-2026	04-September-2026	Remediated	04-September-2026	Successful
6	DDoS_DNS	Low	04-September-2026	04-September-2026	Remediated	04-September-2026	Successful
7	DDoS_DNS	Low	04-September-2026	04-September-2026	Remediated	04-September-2026	Successful
8	DDoS_DNS	Low	04-September-2026	04-September-2026	Remediated	04-September-2026	Successful
9	DDoS_DNS	Low	04-September-2026	04-September-2026	Remediated	04-September-2026	Successful
10	DDoS_DNS	Low	04-September-2026	04-September-2026	Remediated	04-September-2026	Successful
11	DDoS_DNS	Low	04-September-2026	04-September-2026	Remediated	04-September-2026	Successful
12	DDoS_DNS	Low	04-September-2026	04-September-2026	Remediated	04-September-2026	Successful
13	DDoS_DNS	Low	04-September-2026	04-September-2026	Remediated	04-September-2026	Successful
14	DDoS_DNS	Low	04-September-2026	04-September-2026	Remediated	04-September-2026	Successful
15	DDoS_DNS	Low	04-September-2026	04-September-2026	Remediated	04-September-2026	Successful
16	DDoS_DNS	Low	04-September-2026	04-September-2026	Remediated	04-September-2026	Successful

Fig. 5. Remediation Log

VI. CONCLUSION AND FUTURE SCOPE

This paper presented a deep-learning-based framework for detecting and mitigating DDoS attacks

using flow-based network traffic from the CICDDoS2019 dataset, centered on a proposed CNN-BiLSTM hybrid architecture. Nine architectures of increasing structural sophistication were implemented and evaluated under an identical preprocessing pipeline and stratified split, isolating the effect of architectural choice from experimental inconsistency. The proposed CNN-BiLSTM hybrid was the most effective architecture evaluated, achieving 99.69% accuracy, precision, recall, and F1-score, and a ROC-AUC of 0.9992 outperforming every constituent architecture and every simpler baseline. This confirms that combining convolutional local-pattern extraction with bidirectional temporal modelling captures complementary discriminative structure in flow-based network traffic that neither family of architecture captures in isolation. Beyond detection, the framework was extended with a threat-intelligence-driven mitigation module that maps each detected attack family to a risk level, an affected network boundary, and concrete remediation guidance, automatically compiling the results into a structured report. This closes part of the gap between academic detection accuracy and operational usability, providing security analysts with a prioritized, actionable output rather than a flat list of detections. The proposed framework provides a foundation for further research in intelligent DDoS detection and mitigation. Future work can extend the current binary classification approach to multi-class classification, enabling the system to distinguish between individual DDoS attack families and provide attack-specific mitigation strategies. The trained CNN-BiLSTM model can also be integrated into a real-time network monitoring environment using streaming protocols such as IPFIX or NetFlow to support continuous traffic analysis and timely attack detection. Further evaluation on additional intrusion detection datasets, including CICIDS2017 and UNSW-NB15, can be performed to examine the generalization of the model across different traffic characteristics and network environments. Explainable artificial intelligence techniques, such as SHAP and attention-based visualization, can be investigated to identify the network-flow features contributing to individual predictions and improve the interpretability of the detection process. In addition, the threat-intelligence-based mitigation module can be extended toward adaptive and automated response mechanisms capable of recommending or triggering

appropriate actions such as rate limiting, firewall filtering, source blocking, or traffic diversion based on the detected attack type and assigned risk level. The robustness of the proposed model against adversarial and deliberately manipulated traffic can also be investigated through adversarial testing and defensive training. Furthermore, model optimization techniques such as pruning, quantization, and knowledge distillation can be explored to reduce computational and memory requirements, thereby facilitating deployment on resource-constrained edge security devices. Continuous or adaptive learning mechanisms may also be considered to allow the detection model to accommodate evolving DDoS traffic patterns without requiring complete retraining. These future research directions can further develop the proposed framework toward a more generalizable, interpretable, adaptive, and practically deployable DDoS detection and mitigation system.

REFERENCES

- [1] R. P. Janivasya and I. D. A. Rachmawati, "DDoS detection using machine learning approach," *Procedia Computer Science*, vol. 245, pp. 1157–1164, 2024.
- [2] I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani, "Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy," in *Proc. IEEE Int. Carnahan Conf. Security Technology (ICCST)*, 2019.
- [3] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems," in *Proc. Military Communications and Information Systems Conf. (MilCIS)*, 2015.
- [4] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [5] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.
- [6] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. ICISSP*, 2018.
- [7] N. Moustafa and J. Slay, "The evaluation of network anomaly detection systems," *Information Security Journal*, vol. 25, no. 1–3, 2016.
- [8] Y. Mirsky, T. Doitsman, Y. Elovici, and A. Shabtai, "Kitsune: An ensemble of autoencoders for online network intrusion detection," in *Proc. Network and Distributed System Security Symposium (NDSS)*, 2018.
- [9] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, "A survey of network-based intrusion detection data sets," *Computers & Security*, vol. 86, pp. 147–167, 2019.
- [10] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41–50, 2018.
- [11] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, "Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study," *Journal of Information Security and Applications*, vol. 50, Art. no. 102419, 2020.
- [12] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: Techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 20, 2019.
- [13] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, "A deep learning approach for network intrusion detection system," in *Proc. EAI BICT*, 2016.
- [14] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, 2015.
- [15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, vol. 25, 2012.
- [16] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," in *NIPS Workshop on Deep Learning*, 2014.