A Hybrid CNN–BiLSTM Framework with LightGBM Stacking, SDN-Gate, and PSO-Based Threshold Optimization for Enhanced Intrusion Detection in SDN Environments

Maryam Yasmin Ahman
Department of Computer Science, Nile University, Abuja, Nigeria
yasminahman@yahoo.com (corresponding author)

Saleh El-Yakub Abdullahi
Department of Computer Science, Nile University, Abuja, Nigeria
saleh.abdullahi@nileuniversity.edu.ng

Steve Adeshina Adetunji
Department of Computer Engineering, Nile University, Abuja, Nigeria
steve.adeshina@nileuniversity.edu.ng

Received: 2 March 2026 | Revised: 31 March 2026 and 22 April 2026 | Accepted: 23 April 2026

 DOI: https://doi.org/10.48084/etasr.18497

ABSTRACT

Intrusion detection in Software-Defined Networking (SDN) remains challenging due to dynamic control plane traffic and the scarcity of realistic datasets. Conventional Intrusion Detection Systems (IDSs) often struggle to capture diverse SDN-specific threats, including flow rule flooding, topology poisoning, control plane reflection, and other controller-targeted anomalies. This study presents a hybrid Convolutional Neural Network (CNN)–Bidirectional Long Short-Term Memory (BiLSTM) → Light Gradient Boosting Machine (LightGBM) + SDN-Gate with Particle Swarm Optimization (PSO) framework designed to enhance detection accuracy and control plane reliability. A realistic SDN traffic dataset was generated in a GNS3 testbed combining OpenDaylight, Open vSwitch, and multiple Linux hosts, encompassing both general and SDN-specific attacks. The proposed framework employs convolutional and bidirectional recurrent layers for spatial–temporal feature learning, Synthetic Minority Over-sampling Technique–Edited Nearest Neighbor (SMOTE-ENN) for class imbalance mitigation, and LightGBM stacking with PSO-based threshold optimization for calibrated decision fusion. The SDN-Gate, a lightweight LightGBM-based verifier, reevaluates SDN-specific predictions using confidence margins, and verifies and selectively demotes uncertain SDN-specific predictions, thereby reducing false alarms and improving controller-level reliability, and providing a practical foundation for IDS implementations in SDN environments. Experimental results demonstrate 99.80% accuracy and 97.16% Macro-F1 on the full dataset, and 97.22% accuracy and 97.22% Macro-F1 on the Address Resolution Protocol (ARP) and Man-in-the-Middle (MITM) attacks subset, outperforming baseline deep and shallow learning models. Overall, the proposed framework provides a reliable and explainable approach to improving and strengthening the security of SDN networks in real-world settings.

Keywords-Software-Defined Networking (SDN); Intrusion Detection Systems (IDSs); Deep Learning (DL); CNN–BiLSTM; Particle Swarm Optimization (PSO)

I. INTRODUCTION

Software-Defined Networking (SDN) separates the control and data planes, enabling centralized policy management and programmability across data center, cloud, and enterprise infrastructures. This flexibility enhances visibility and automation but also introduces new security risks, particularly when weaknesses in the control plane can affect the entire network.

SDN operates through three logical layers: the application plane, where network policies and services such as Intrusion Detection Systems (IDSs), Access Control Lists (ACLs), and Quality of Service (QoS) are defined; the control plane, where the controller maintains topology, computes paths, and manages southbound and northbound communications; and the data plane, comprising switches and routers that forward packets based on controller rules [1]. This layered abstraction simplifies automation and security enforcement while enabling flexible management; yet it also centralizes risk, making intrusion detection essential for safeguarding control-to-data plane interactions. IDSs are therefore essential to identify abnormal behaviors; however, conventional signature-based methods struggle with stealthy or low-rate attacks that resemble normal traffic. Deep Learning (DL) techniques offer improved adaptability by automatically learning spatial–temporal traffic patterns [2], yet their application to SDN remains constrained by outdated or non-SDN datasets, limited SDN-specific attack coverage, unbalanced traffic distributions, and under-explored hybrid architectures [3-5].

DL has been applied widely to intrusion detection, with Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM)/Bidirectional Long Short-Term Memory (BiLSTM) families attracting attention for their ability to model spatial and temporal structure in traffic. Recent SDN-oriented studies explore hybrids like CNN–BiLSTM, attention modules, and in some cases transformers and ensembles, aiming to raise accuracy on diverse attack types while reducing overfitting. In this context, authors in [6] proposed a hybrid DL-powered SDN-based intrusion detection architecture for Internet of Things (IoT) environments, demonstrating the effectiveness of hybrid DL models for detecting cyber threats in SDN-enabled networks.

Authors in [7] evaluated multiple deep models for SDN IDSs and found that an attention-based CNN–BiLSTM achieved the best results (98.03% attack detection), outperforming plain CNN–BiLSTM (97.26%), CNN, and classic vision backbones (LeNet-5, AlexNet). The attention layer helped the model emphasize salient flow features, improving separation between benign and malicious traffic on the InSDN dataset. The study highlights attention as a useful augmentation for SDN traffic where patterns are subtle and context dependent.

Authors in [8] combined CNN and LSTM for SDN intrusion detection on InSDN, using L2 regularization and dropout to prevent overfitting. The hybrid captured spatial cues (via convolutions) and temporal dependencies (via LSTM) and achieved about 96.32%, accuracy surpassing individual CNN/LSTM variants. The regularization choices also improved resilience to previously unseen (zero-day-like) behaviors relative to a plain CNN.

Authors in [9] focused on Denial of Service (DoS)/Distributed Denial of Service (DDoS) in SDN and compared three strategies: a voting ensemble achieving 97.4% accuracy, a stacking ensemble achieving 99% accuracy as the best, and a CNN–LSTM hybrid with 98.51% accuracy. The study suggests that meta-learning with stacked models can edge out single deep architectures for volatile attack families, whereas the CNN–LSTM hybrid remains a strong, lower-complexity alternative.

Authors in [10] targeted slow DDoS in SDN, an attack profile that stresses temporal modeling. Their CNN–LSTM on a custom SDN dataset delivered >99% across reported metrics and outperformed Multilayer Perceptron (MLP) and one-class Support Vector Machine (SVM). Results indicate that combining convolutional feature extractors with sequence modeling is well-suited for low-rate, long-horizon anomalies typical of controller-focused attacks.

Hybrid CNN–LSTM/BiLSTM models consistently outperform single architectures, whereas attention mechanisms improve minority-class detection and ensembles enhance robustness against evolving attacks. However, performance remains strongly dependent on dataset diversity and balance, particularly adequate representation of network and control-plane behaviors in SDN environments, where rare attacks must be captured.

This study aims to develop and evaluate a robust and deployment-oriented intrusion detection framework for SDN environments. Specifically, it seeks to enhance the detection of both conventional and SDN-specific threats using realistic traffic data. The study further investigates the effectiveness of combining DL and optimization techniques under imbalanced and complex traffic conditions. Additionally, it evaluates the framework's ability to maintain high detection accuracy while minimizing false alarms within the SDN control plane.

To address these challenges, this work proposes a deployment-oriented hybrid intrusion detection framework integrating realistic traffic generation, spatial–temporal learning, and optimization-driven refinement. The main contributions are: (1) a realistic SDN dataset generated in a GNS3 environment with OpenDaylight, Open vSwitch, and Linux hosts covering both conventional and SDN-specific attacks; (2) a CNN–BiLSTM architecture for spatial–temporal modeling of SDN traffic; (3) leak-free training using the Synthetic Minority Over-sampling Technique–Edited Nearest Neighbor (SMOTE-ENN), applied exclusively to the training split for class imbalance mitigation; and (4) a hybrid decision framework combining Light Gradient Boosting Machine (LightGBM) stacking and an inference-time SDN-Gate that verifies and selectively demotes uncertain SDN-specific predictions, with Particle Swarm Optimization (PSO)-based threshold optimization, to improve detection accuracy and reduce controller-plane false alarms.

This study is significant to the scientific community as it addresses critical gaps in SDN intrusion detection by providing both a realistic dataset and a reproducible evaluation framework. The publicly available SDN-IDPS dataset offers a valuable benchmark for future research, particularly in assessing SDN-specific attack scenarios. In addition, the proposed framework demonstrates a practical approach to improving detection robustness in programmable network environments. These contributions can support the development of more reliable and scalable security solutions for modern SDN deployments.

II. RESEARCH METHODOLOGY

This study follows an experimental research design focused on building a realistic SDN testbed, generating both benign and malicious traffic, and developing a hybrid DL-based intrusion detection framework.

The overall workflow comprised four sequential phases: data collection, preprocessing and feature engineering, model design, and optimization & evaluation.

A. Data Collection and Environment Setup

A three-tier SDN architecture was implemented in GNS3 [11], emulating real-world controller switch interactions.

- Control plane: OpenDaylight [12] controller on Ubuntu 24.04.1 LTS.
- Data plane: Five Open vSwitch [13] instances, each with 15 ports supporting host and inter-switch connections.
- Application layer: An FTP server providing file transfer services and OWASP Juice Shop [14] web application providing legitimate HTTP/HTTPS traffic.

Sixty-nine Debian and Lubuntu hosts generated benign traffic, whereas a dedicated Kali Linux node launched attacks. The setup ran on a Windows Server 2016 host (Intel Xeon Gold 6138, 20 cores @ 2.00 GHz, 96 GB RAM and 2 TB storage).

1) Traffic Generation

A realistic dataset, hereafter referred to as the SDN-IDPS dataset, was generated within a controlled GNS3 environment [11] consisting of an OpenDaylight controller [12], five Open vSwitch instances [13], and 70 Linux-based hosts. Benign traffic included routine web browsing, FTP and DNS operations, email, and SSH sessions. The OWASP Juice Shop [14] web application was used to simulate user activity, creating realistic HTTP/HTTPS workloads while also serving as the target for web-based attacks.

Malicious traffic was introduced in phases representing reconnaissance, exploitation, and post-exploitation behaviors across all SDN layers. Attacks were executed with standard penetration testing tools such as hping3 [15], SlowHTTPTest [16], GoldenEye [17], Hydra [18], Medusa [19], Nmap [20], Masscan [21], sdnpwn [22], Sqlmap [23], Burp Suite [24], Bettercap [25], and Metasploit [26], supplemented by custom Python/Scapy scripts for SDN-specific attacks. All captured traffic was stored as PCAP files using Wireshark [27]. Detailed dataset statistics and attack distributions are provided in the publicly available repository [28].

B. Data Preprocessing and Feature Engineering

PCAP files captured were converted into bidirectional flow records using CICFlowMeter [29] for the unified dataset, which produced CSV outputs with standard flow statistics such as packet counts, durations, and byte rates. For control-plane captures, such as Address Resolution Protocol (ARP) spoofing and Man-in-the-Middle (MITM), where CICFlowMeter's IP-centric parsing was insufficient, TShark [30] was used to extract Layer-2 and ARP fields from packet captures between the OpenDaylight controller, Open vSwitch, and the Kali Linux attacker node. The dataset contained eight classes: Normal, DoS, DDoS, Web Attack, Reconnaissance, R2L (credential access), Malware, and SDN Specific, comprising 84 flow-based features. Non-statistical identifiers (e.g., Flow ID and IP addresses) were removed to prevent leakage, leaving 77 numeric and behavioral features. Missing values were cleaned, categorical fields encoded, and all attributes were standardized using Z-score normalization.

For the SDN-specific ARP spoofing and MITM subset, 29 temporal and directional features were retained to represent spoofing and interception behaviors.

To handle class imbalance, SMOTE-ENN was applied exclusively to the training split (80/20 partition), generating synthetic minority samples and pruning noisy boundary instances.

C. Model Design

The proposed framework follows a hybrid architecture comprising a CNN–BiLSTM backbone, a LightGBM stacking classifier, an inference-time SDN-Gate, and PSO-optimized decision thresholds, designed to capture spatial–temporal traffic behaviors while enhancing SDN-specific detection reliability. The CNN–BiLSTM backbone serves as a deep feature encoder: the CNN extracts local spatial patterns from flow features, whereas the BiLSTM models bidirectional temporal dependencies, producing 128-dimensional behavioral embeddings. The resulting embeddings are combined with supplementary decision features and forwarded to a LightGBM stacking classifier for final multi-class decision learning. An inference-time SDN-Gate, implemented as a lightweight LightGBM-based verifier, reevaluates SDN-specific predictions using confidence margins and verifies and selectively demotes uncertain outputs to reduce controller-plane false positives.

D. Model Optimization and Evaluation

The CNN–BiLSTM backbone was trained using the Adam optimizer. PSO was subsequently applied during model refinement to optimize class-specific decision thresholds for improved Macro-F1 balance. During inference, the LightGBM stacking classifier first produces a multi-class prediction, after which the LightGBM-based SDN-Gate verifies and selectively demotes uncertain SDN-specific predictions using confidence and probability margin checks, filtering uncertain controller-plane alerts while maintaining recall for legitimate anomalies. Model performance was assessed using standard classification metrics: Accuracy (1), Precision (2), Recall (3), F1-score (4), and Macro-F1 (5). Confusion metrics were reported only for the final hybrid configuration to avoid repetition across intermediate models.

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN} \quad (1)$$

$$\text{Precision} = \frac{TP}{TP+FP} \quad (2)$$

$$\text{Recall} = \frac{TP}{TP+FN} \quad (3)$$

$$\text{F1}-\text{score} = \frac{2PR}{P+R} \quad (4)$$

The Macro-F1 score, representing the unweighted mean of all class-wise F1-scores, is given by:

$$\text{Macro-F1} = \frac{1}{C}\sum_{i=1}^{C} \text{F1}-\text{score}_i \quad (5)$$

Here, TP, TN, FP, and FN denote true positives, true negatives, false positives, and false negatives, respectively. All preprocessing, resampling, and PSO tuning were restricted to the training and validation partitions, ensuring that the test set remained completely untouched for unbiased evaluation.

III. RESULTS AND DISCUSSION

A. Overview

The proposed CNN–BiLSTM → LightGBM + SDN-Gate (PSO) framework was evaluated on the SDN-IDPS dataset covering both conventional and control plane attacks. The CNN–BiLSTM backbone learned spatial–temporal traffic patterns, with the resulting embeddings passed to a LightGBM classifier for multi-class prediction. PSO was used to refine decision thresholds for improved class-wise balance. An inference-time SDN-Gate then verifies and selectively demotes uncertain SDN-specific predictions using confidence margins to reduce controller-plane false alarms. Together, these components improve detection reliability across both data and control-plane traffic in SDN environments.

B. Experimental Setup

All experiments were conducted in Python 3.12 [31] on Google Colab Pro (A100 GPU) [32] using TensorFlow 2.19 [33]/Keras [34] for DL and LightGBM 4.6 [35] for stacking and the SDN-Gate. An 80/20 train/validation split with an untouched test set was adopted.

The CNN–BiLSTM backbone was trained for ≤ 50 epochs (batch size 128, Adam 1×10^{-4}) with early stopping on validation Macro-F1; SMOTE-ENN was applied exclusively to the training split for class imbalance mitigation.

After training, PSO was applied to optimize per-class decision thresholds for improved class-wise balance. All optimization was performed using the validation split, whereas the test set remained completely unseen during training and tuning. During inference, the LightGBM stacking classifier first produces multi-class predictions, after which a lightweight LightGBM-based SDN-Gate verifies and selectively demotes uncertain SDN-specific predictions using probability margin checks to reduce false positives.

C. Ablation Experiments

To quantify the contribution of each architectural refinement, six model variants ranging from a baseline BiLSTM to the full CNN–BiLSTM → LightGBM + SDN-Gate (PSO) framework were evaluated under identical experimental conditions. Each enhancement (convolutional feature extraction, stacking, PSO-based threshold calibration, and gating) progressively improved Macro-F1 performance. Comparative Macro-F1 performance across ablation stages is illustrated in Figure 1, whereas the detailed stage-wise results are summarized in Table I. A corresponding staged comparison on the ARP and MITM SDN-specific subset is provided in Table II.

D. Performance Analysis

As shown in Table I, overall accuracy increased from 97.68% for BiLSTM to 99.80% for the proposed framework, whereas Macro-F1 rose from 77.31% to 97.16%. DoS and DDoS traffic maintained near-perfect precision and recall (>99%), confirming that the CNN–BiLSTM backbone effectively captured large-scale spatial–temporal patterns. Reconnaissance and Web Attack flows also remained highly separable with F1-scores above 98%.

The largest improvements were observed for minority classes: Malware and R2L F1-scores increased from 39% to 90.61% and from 77.16% to 95.67%, respectively, following the introduction of stacking and PSO-based threshold calibration. The greatest improvement occurred in SDN-specific detection: the baseline BiLSTM achieved only 9.5% F1-score, but integrating the inference-time SDN-Gate improved performance to approximately 93%. The SDN-Gate, implemented as a lightweight LightGBM-based module, operates solely during inference and verifies and selectively demotes uncertain SDN-specific predictions through probability margin checks, thereby confirming controller-plane alerts and reducing false positives. Normal traffic also improved from 97.9% to 99.6%, indicating stable class boundaries and reduced overfitting.

To further examine control-plane behavior, the model was evaluated on an SDN-specific subset containing ARP spoofing and MITM attacks, as shown in Table II. The baseline BiLSTM struggled with Layer-2 temporal dependencies, achieving 66.67% accuracy. Introducing convolutional layers enabled spatial–temporal feature extraction and increased accuracy to 97.22%. Performance plateaued after CNN–BiLSTM, indicating that the learned representations were already linearly separable. The SDN-Gate and PSO configurations did not yield additional performance gains on this smaller subset, reflecting the simplicity and clear separability of the binary classification task. However, the LightGBM stacking classifier maintained stable performance across the final configurations (97.22% accuracy and Macro-F1). Given that the subset forms a binary classification task (Normal vs SDN-specific) with clearly separable patterns, configurations incorporating SDN-Gate did not improve performance in this setting, whereas the LightGBM-based configurations remained stable.

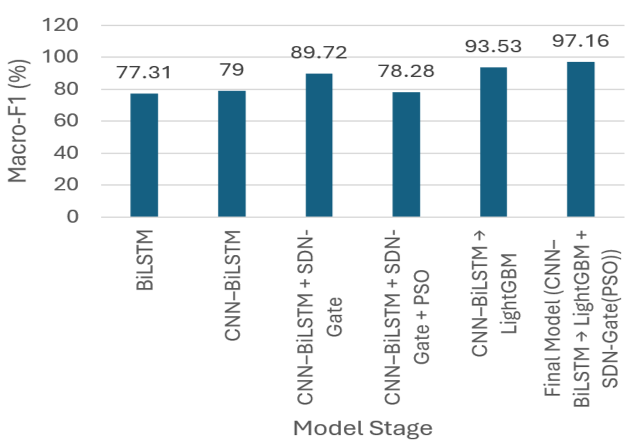

Fig. 1. Comparative Macro-F1 performance across ablation stages.

TABLE I. COMPARATIVE PERFORMANCE OF MODEL VARIANTS ACROSS ALL TRAFFIC CLASSES

Attack class	Metric (%)	BiLSTM	CNN–BiLSTM	CNN–BiLSTM + SDN-Gate	CNN–BiLSTM + SDN-Gate + PSO	CNN–BiLSTM → LightGBM	CNN–BiLSTM → LightGBM + SDN-Gate (PSO)
DDoS	Precision	99.25	99.00	99.32	99.37	99.90	99.85
	Recall	99.84	100.00	99.85	99.79	99.88	99.89
	F1-score	99.54	100.00	99.59	99.58	99.89	99.87
DoS	Precision	99.83	100.00	99.86	99.79	99.91	99.89
	Recall	95.69	97.00	98.61	97.08	99.72	99.78
	F1-score	97.72	98.00	99.23	98.42	99.82	99.84
Malware	Precision	24.89	36.00	38.11	31.34	75.00	86.67
	Recall	90.79	92.00	92.06	91.75	95.24	94.92
	F1-score	39.07	51.00	53.90	46.73	83.92	90.61
Normal	Precision	97.53	99.00	96.31	99.21	99.51	99.55
	Recall	98.24	98.00	98.78	98.18	99.68	99.75
	F1-score	97.89	98.00	97.53	98.69	99.60	99.65
R2L	Precision	63.01	63.00	66.12	62.35	80.68	91.90
	Recall	99.50	100.00	99.75	99.63	99.75	99.75
	F1-score	77.16	77.00	79.53	76.70	89.21	95.67
Reconnaissance	Precision	99.70	100.00	99.86	94.59	99.88	99.79
	Recall	99.41	99.00	99.41	99.44	99.63	99.64
	F1-score	99.56	100.00	99.64	96.96	99.75	99.71
SDN Specific	Precision	5.01	7.00	89.98	10.41	62.89	92.38
	Recall	98.79	97.00	97.45	91.42	99.73	94.24
	F1-score	9.53	13.00	93.56	18.69	77.14	93.30
Web Attack	Precision	97.24	94.00	90.70	83.06	98.17	97.74
	Recall	98.80	99.00	99.33	99.42	99.64	99.59
	F1-score	98.01	96.00	94.82	90.50	98.90	98.66
Overall accuracy (%)		97.68	98.00	99.13	98.32	99.77	99.80
Macro-F1 (%)		77.31	79.00	89.72	78.28	93.53	97.16

TABLE II. COMPARATIVE PERFORMANCE OF MODELS ON ARP AND MITM SDN-SPECIFIC SUBSET

Model stage	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Macro-F1 (%)
BiLSTM	66.67	74.11	66.67	63.88	63.88
CNN–BiLSTM	97.22	97.37	97.22	97.22	97.22
CNN–BiLSTM + SDN-Gate	61.11	78.13	61.11	54.18	54.18
CNN–BiLSTM + SDN-Gate + PSO	55.56	76.47	55.56	44.62	44.62
CNN–BiLSTM → LightGBM	97.22	97.37	97.22	97.22	97.22
CNN–BiLSTM → LightGBM + PSO	97.22	97.37	97.22	97.22	97.22

From a deployment perspective, the proposed framework introduces minimal computational overhead during inference, making it suitable for practical network environments. Resource-intensive processes, including SMOTE-ENN balancing and PSO-based threshold optimization, are performed entirely offline during the training phase and therefore impose no runtime cost on the controller. During inference, the pipeline executes three lightweight sequential operations: CNN–BiLSTM feature extraction, LightGBM multi-class classification, and conditional SDN-Gate verification, the latter activating only for SDN-specific predictions.

While the CNN–BiLSTM component contributes the highest per sample computational cost, it remains bounded by a fixed model structure, whereas LightGBM inference is comparatively efficient. In practice, the detection pipeline may be deployed as an embedded controller application or as an external analysis service, depending on latency and resource constraints. Although the framework is designed to support near real-time operation, inference latency may increase under very high traffic volumes or large-scale deployments, and precise latency benchmarking under varying SDN topologies and traffic conditions remains an important direction for future work.

IV. CONCLUSION

This study presented a hybrid Convolutional Neural Network (CNN)–Bidirectional Long Short-Term Memory (BiLSTM) → Light Gradient Boosting Machine (LightGBM) + SDN-Gate with Particle Swarm Optimization (PSO) framework for intrusion detection in Software-Defined Networking (SDN) environments. Experimental results show that the proposed framework achieves 99.80% accuracy and 97.16% Macro-F1 on the full SDN-IDPS dataset, and 97.22% accuracy and Macro-F1 on the Address Resolution Protocol

(ARP) and Man-in-the-Middle (MITM) SDN-specific subset, indicating effective detection of both conventional and SDN-specific attack patterns. These results demonstrate that combining spatial–temporal learning with data balancing and optimization-driven stacking improves classification performance in complex and imbalanced SDN traffic scenarios. The framework's inference-only deployment design, with resource-intensive processes confined to the offline training phase, further supports practical integration into production SDN environments.

While the evaluation is conducted within a controlled SDN testbed, certain limitations should be acknowledged. The dataset, although realistic, is generated within an emulated SDN environment and may not fully capture the variability and scale of real-world deployments. Additionally, class imbalance and the controlled nature of attack scenarios, while diverse, may limit generalization to highly dynamic, previously unseen, or stealthy attack patterns. Despite these constraints, the findings highlight the potential of the proposed framework for practical deployment in programmable networks; however, validation in real-world operational environments remains an important direction for future work.

Future work will extend the dataset and evaluation to larger and more diverse environments, including multi-controller and hybrid cloud SDN architectures, while improving adaptability to evolving attack behaviors. In addition, further investigation into inference latency and deployment scalability will be conducted to enhance practical applicability in real-world SDN environments.

DECLARATION OF COMPETING INTERESTS

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

ACKNOWLEDGMENT

The authors acknowledge that this research was self-funded and conducted using available institutional and personal resources. No external funding was received for this study.

DATA AVAILABILITY

The SDN-IDPS dataset and associated supplementary materials, including feature descriptions and class distribution details, are publicly available on Zenodo and can be accessed in [28].

REFERENCES

[1] D. Kreutz, F. M. V. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, Jan. 2015, https://doi.org/10.1109/JPROC.2014.2371999.

[2] N. P. Mwanza and J. Kalita, "Detecting DDoS Attacks in Software Defined Networks Using Deep Learning Techniques: A Survey," *International Journal of Network Security*, vol. 25, no. 2, pp. 360–376, Mar. 2023.

[3] M. Mittal, K. Kumar, and S. Behal, "Deep learning approaches for detecting DDoS attacks: a systematic review," *Soft Computing*, vol. 27, no. 18, pp. 13039–13075, Sept. 2023, https://doi.org/10.1007/s00500-021-06608-1.

[4] M. S. Elsayed, N.-A. Le-Khac, and A. D. Jurcut, "InSDN: A Novel SDN Intrusion Dataset," *IEEE Access*, vol. 8, pp. 165263–165284, 2020, https://doi.org/10.1109/ACCESS.2020.3022633.

[5] N. Ahmed *et al.*, "Network Threat Detection Using Machine/Deep Learning in SDN-Based Platforms: A Comprehensive Analysis of State-of-the-Art Solutions, Discussion, Challenges, and Future Research Direction," *Sensors*, vol. 22, no. 20, Oct. 2022, Art. no. 7896, https://doi.org/10.3390/s22207896.

[6] T. Panse, V. Gaddam, B. S. Manthina, H. R. Battu, P. Sunitha, and V. Sailaja, "A Hybrid Deep Learning-Powered SDN-Based Intrusion Detection Architecture for Cognitive IoT Security," *Engineering, Technology & Applied Science Research*, vol. 15, no. 5, pp. 27495–27501, Oct. 2025, https://doi.org/10.48084/etasr.12564.

[7] R. B. Said and I. Askerzade, "Attention-Based CNN-BiLSTM Deep Learning Approach for Network Intrusion Detection System in Software Defined Networks," in *2023 5th International Conference on Problems of Cybernetics and Informatics*, Baku, Azerbaijan, 2023, pp. 1–5, https://doi.org/10.1109/PCI60110.2023.10325985.

[8] M. Abdallah, N. An Le Khac, H. Jahromi, and A. Delia Jurcut, "A Hybrid CNN-LSTM Based Approach for Anomaly Detection Systems in SDNs," in *Proceedings of the 16th International Conference on Availability, Reliability and Security*, Vienna, Austria, 2021, pp. 1–7, https://doi.org/10.1145/3465481.3469190.

[9] M. Afroj, K. M. S. Rifat, and Md. S. Rahman, "Enhanced Detection of DoS/DDoS Attacks in SDN Using Ensemble and Hybrid CNN-LSTM Models," in *2024 IEEE International Conference on Computing, Applications and Systems*, Cox's Bazar, Bangladesh, 2024, pp. 1–6, https://doi.org/10.1109/COMPAS60761.2024.10796601.

[10] B. Nugraha and R. N. Murthy, "Deep Learning-based Slow DDoS Attack Detection in SDN-based Networks," in *2020 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, Leganes, Spain, 2020, pp. 51–56, https://doi.org/10.1109/NFV-SDN50289.2020.9289894.

[11] "GNS3 | The software that empowers network professionals." GNS3. https://www.gns3.com/.

[12] "OpenDaylight: Automating networks of any size & scale." OpenDaylight. https://www.opendaylight.org/.

[13] "Open vSwitch: Production Quality, Multilayer Open Virtual Switch." Openvswitch. https://www.openvswitch.org/.

[14] "OWASP Juice Shop | OWASP Foundation." Owasp. https://owasp.org/www-project-juice-shop/.

[15] "hping3 | Kali Linux Tools." Kali Linux. https://www.kali.org/tools/hping3/.

[16] "slowhttptest | Kali Linux Tools." Kali Linux. https://www.kali.org/tools/slowhttptest/.

[17] "goldeneye | Kali Linux Tools." Kali Linux. https://www.kali.org/tools/goldeneye/.

[18] "hydra | Kali Linux Tools." Kali Linux. https://www.kali.org/tools/hydra/.

[19] "medusa | Kali Linux Tools." Kali Linux. https://www.kali.org/tools/medusa/.

[20] "nmap | Kali Linux Tools." Kali Linux. https://www.kali.org/tools/nmap/.

[21] "masscan | Kali Linux Tools." Kali Linux. https://www.kali.org/tools/masscan/.

[22] D. Smyth, "smythtech/sdnpwn." Apr. 07, 2026. [Online]. Available: https://github.com/smythtech/sdnpwn.

[23] "sqlmap | Kali Linux Tools." Kali Linux. https://www.kali.org/tools/sqlmap/.

[24] "burpsuite | Kali Linux Tools." Kali Linux. https://www.kali.org/tools/burpsuite/.

[25] "bettercap | Kali Linux Tools." Kali Linux. https://www.kali.org/tools/bettercap/.

[26] "metasploit-framework | Kali Linux Tools." Kali Linux. https://www.kali.org/tools/metasploit-framework/.

[27] "Wireshark • Go Deep." Wireshark. https://www.wireshark.org/.

[28] M. Y. Ahman, "SDN_IDPS: A Realistic Software-Defined Networking Intrusion Detection Dataset with SDN-Specific Attack Modeling." Zenodo, Mar. 30, 2026, https://doi.org/10.5281/zenodo.19335462.

[29] "CICFlowMeter (formerly ISCXFlowMeter)." Canadian Institute for Cybersecurity | UNB. https://www.unb.ca/cic/research/applications.html.

[30] "tshark(1) Manual Page." Wireshark. https://www.wireshark.org/docs/man-pages/tshark.html.

[31] "Python Release Python 3.12.11." Python. https://www.python.org/downloads/release/python-31211/.

[32] "Colab." Google for Developers. https://developers.google.com/colab.

[33] "TensorFlow: An end-to-end platform for machine learning." TensorFlow. https://www.tensorflow.org/.

[34] "Keras: Deep Learning for humans." Keras. https://keras.io/.

[35] "Welcome to LightGBM's documentation! — LightGBM 4.6.0.99 documentation." LightGBM. https://lightgbm.readthedocs.io/en/latest/.