International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(7s) 2026 pp. 515-526
https://doi.org/10.70917/ijcisim-2026-3113
Article

A Hybrid Deep Learning and Machine Learning Model for Intelligent Cyber Threat Detection in Smart Networks

Rajesh Yadav[1*], Dinesh Kumar[2], Sanjeev Kumar[3], Ruchi Singhal[4], Prashant Kumar[5], Munish Kumar[6]

[1]Department of Computer Science, University of Southampton Delhi, India
Email: Rajesh.yadav@southampton.ac.uk
[2]Department of Computer Science and Applications, Vivekananda Global University, Jaipur, Rajasthan, India, Email: Dineshkumarjune10@gmail.com
[3]Department of Computer Science and Engineering, Maharaja Agrasen Institute of Technology, Rohini, Sec 22, New Delhi, India, Email: sanjeevkumar@mait.ac.in
[4]Department of Information Technology, Jagannath International Management School, Vasant Kunj, New Delhi, India, Email ruchi.singhal75@gmail.com
[5]Department of Information Technology, Jagannath International Management School, Kalkaji, New Delhi, India, Email: prashantkumar@yahoo.com
[6]Independent Researchers, Sirsa, Haryana, India. *Email: munish2012@gmail.com*
[*]Corresponding Author: Rajesh Yadav

Abstract: The rapid expansion of smart networks, encompassing the Internet of Things (IoT), software-defined networking (SDN), and 5G-enabled edge infrastructure, has dramatically increased the attack surface available to malicious actors, while simultaneously producing high-velocity, heterogeneous traffic that traditional signature-based intrusion detection systems struggle to analyze in real time. This paper proposes a Hybrid Deep Learning and Machine Learning (DL-ML) framework for intelligent cyber threat detection that fuses a Convolutional Neural Network combined with a Bidirectional Long Short-Term Memory (CNN-BiLSTM) branch, which captures spatial and temporal traffic patterns, with a gradient-boosted ensemble branch (XGBoost/Random Forest), which captures statistical flow-level signatures. The outputs of both branches are combined through a weighted feature-fusion and ensemble layer that produces a unified threat classification and severity score. The framework was evaluated on a large-scale smart-network intrusion dataset comprising over 1.8 million labeled flow records spanning six traffic classes: normal, DDoS, botnet, port scanning, malware communication, and spoofing. Experimental results show that the proposed hybrid model achieves 98.8% accuracy, 96.4% precision, 95.6% recall, and a 96.0% F1-score, exceeding the strongest individual baseline (LSTM) by 3.7 percentage points in F1-score and achieving an AUC of 0.992.

Keywords: Cyber Threat Detection, Deep Learning, Machine Learning, Hybrid Model, Smart Networks, IoT Security, CNN-BiLSTM, XGBoost, Intrusion Detection System, Network Security

1. INTRODUCTION

Smart networks, comprising Internet of Things (IoT) devices, software-defined networking (SDN) infrastructure, and increasingly autonomous edge-computing nodes, have become foundational to modern critical infrastructure, smart cities, industrial automation, and consumer connectivity. This proliferation of interconnected, often resource-constrained devices has expanded the cyber-attack surface dramatically, giving rise to large-scale distributed denial-of-service (DDoS) campaigns, botnet propagation, port scanning reconnaissance, malware command-and-control communication, and address-spoofing attacks that can compromise entire network segments

within seconds [1-2], [21-22]. Traditional signature-based intrusion detection systems (IDS), while computationally efficient, are fundamentally reactive: they can only detect previously catalogued attack signatures and are therefore ineffective against zero-day exploits and adversarially obfuscated traffic patterns [3], [15].

Machine learning (ML) techniques, including Support Vector Machines, Random Forests, and gradient-boosted ensembles, have been widely applied to network intrusion detection due to their ability to learn statistical decision boundaries directly from labeled flow data, generalizing beyond fixed signatures [4], [16-17], [22-23]. However, purely statistical ML models typically rely on hand-engineered, flow-level features and struggle to capture the fine-grained temporal dependencies and spatial correlations present in raw packet sequences, particularly for attacks that unfold gradually over time, such as low-rate DDoS or stealthy botnet beaconing [5], [18], [24]. Deep learning (DL) architectures, particularly Convolutional Neural Networks (CNNs) and recurrent architectures such as Long Short-Term Memory (LSTM) networks, address this limitation by automatically learning hierarchical spatial and temporal representations directly from raw or lightly processed traffic data [6-7], [19], [25]. Nevertheless, DL models are computationally expensive, require substantial labeled training data, and often underperform simpler statistical models on well-structured, low-dimensional tabular features that are already highly informative for certain attack types [8], [20], [26].

This complementary strength-and-weakness profile motivates a hybrid approach. This paper proposes a Hybrid Deep Learning and Machine Learning framework that combines a CNN-BiLSTM branch, which captures spatio-temporal patterns from sequential traffic windows, with a gradient-boosted ensemble branch, which captures discriminative statistical flow-level features, fusing both through a learned weighted ensemble layer. Prior hybrid intrusion detection studies have explored similar combinations [9-12], [27-28] but frequently rely on simple concatenation or static averaging rather than an adaptively weighted fusion mechanism, and few studies jointly evaluate performance, computational overhead, and per-attack-class behavior on large, multi-class smart-network datasets, a gap this paper addresses directly.

The key contributions of this paper are as follows:

1. A hybrid DL-ML architecture combining CNN-BiLSTM spatio-temporal feature learning with gradient-boosted statistical ensemble learning for smart-network threat detection.
2. An adaptive, weighted feature-fusion layer that learns the relative contribution of the deep and shallow branches per traffic class, rather than relying on fixed-weight or simple-averaging ensembles.
3. A comprehensive empirical evaluation on a large-scale, six-class smart-network intrusion dataset, benchmarked against five baseline models across accuracy, precision, recall, F1-score, and AUC.
4. An ablation study isolating the contribution of each architectural branch and the fusion mechanism, together with a global feature-importance analysis of the most discriminative network-traffic indicators.

The remainder of this paper is organized as follows. Section 2 reviews related work on machine learning, deep learning, and hybrid approaches to network intrusion detection. Section 3 formalizes the problem statement. Section 4 describes the proposed hybrid framework in detail. Section 5 presents the experimental setup, dataset, and evaluation protocol. Section 6 reports and discusses the results, including ablation and feature-importance analysis. Section 7 discusses limitations, and Section 8 concludes the paper with directions for future work.

2. RELATED WORK

2.1 Signature-Based and Classical Machine Learning Approaches

Early network intrusion detection systems relied on signature matching against known attack patterns, an approach that remains fast and precise for previously seen threats but is inherently blind to novel or obfuscated attacks [3], [15]. To move beyond static signatures, researchers applied classical machine learning to flow-level features. Support Vector Machines and Naive Bayes classifiers were among the first statistical models applied to benchmark intrusion datasets, demonstrating reasonable discrimination but sensitivity to feature scaling and class imbalance [4], [16]. Random Forest and gradient boosting frameworks such as XGBoost later improved robustness and accuracy on structured flow features, becoming widely used baselines for network intrusion detection due to their strong tabular performance and relatively low training cost [17]. However, these models generally treat each flow record as an independent, fixed-length feature vector and cannot directly exploit sequential or spatial dependencies across packets or time windows.

2.2 Deep Learning for Network Intrusion Detection

Deep learning architectures have been increasingly applied to intrusion detection to automatically learn hierarchical representations from raw or minimally processed traffic. Convolutional Neural Networks have been used to treat packet byte sequences or flow feature matrices as image-like inputs, learning local spatial patterns indicative of malicious payloads [6], [18]. Recurrent architectures, particularly LSTM and Gated Recurrent Unit (GRU) networks, have been applied to model the temporal evolution of network flows, capturing multi-step attack behaviors such as slow-rate DDoS ramp-up or periodic botnet beaconing that are difficult to detect from single-flow snapshots [7], [19]. Hybrid CNN-LSTM architectures that combine spatial and temporal feature learning within a single deep model have shown further improvements, particularly for attacks with both structural and temporal signatures, such as coordinated botnet traffic [5], [8].

2.3 Hybrid Deep Learning and Machine Learning Frameworks

A growing body of work has explored combining deep and shallow learning models for intrusion detection. Vinayakumar et al. proposed deep learning approaches for intrusion detection benchmarked across multiple datasets, establishing strong DL baselines for comparison with classical ML [9]. Other studies have proposed stacked ensembles combining CNN-derived features with gradient-boosted classifiers, typically using simple concatenation of feature vectors prior to a shallow classifier [10], [11]. Moustafa and Slay contributed comprehensive intrusion detection benchmark datasets that have become standard evaluation platforms for both DL and ML models in this domain [20]. Ferrag et al. surveyed deep learning approaches for cyber-security intrusion detection, noting that most hybrid proposals in the literature use static or unweighted fusion strategies and rarely evaluate per-class performance in smart-network settings characterized by highly imbalanced and heterogeneous device traffic [12]. This paper builds directly on this literature by introducing an adaptively weighted fusion mechanism and evaluating it comprehensively across six attack classes representative of modern smart-network threats.

2.4 Smart Network and IoT-Specific Security Challenges

Smart networks introduce unique challenges beyond conventional enterprise networks, including highly constrained device compute and power budgets, heterogeneous communication protocols (MQTT, CoAP, Zigbee), and dynamic, software-defined topologies that can be reconfigured in real time [1-2], [13], [14]. These characteristics necessitate detection frameworks that are both accurate and computationally efficient enough to operate at or near the network edge, a design consideration explicitly addressed in the proposed framework's evaluation of inference latency alongside detection accuracy.

3. PROBLEM STATEMENT AND MOTIVATION

Let a network flow record be represented as a feature vector x_t at time t, comprising statistical attributes (packet count, byte count, flow duration, protocol type) and a corresponding short sequence of packet-level observations S_t = {p_1, p_2, ..., p_n} within a sliding time window. The threat detection task is formulated as a multi-class classification problem: given x_t and S_t, learn a function f(x_t, S_t) -> c, where c belongs to the label set {Normal, DDoS, Botnet, Port Scan, Malware, Spoofing}, that assigns each flow to its correct traffic class with high precision and low detection latency.

This formulation surfaces three central challenges. First, severe class imbalance: benign traffic vastly outnumbers any individual attack class, biasing naive classifiers toward the majority class and requiring targeted loss weighting or resampling. Second, heterogeneous signal structure: some attack types (e.g., port scanning) are best identified from statistical flow features, while others (e.g., coordinated botnet beaconing) are best identified from temporal packet sequences, meaning no single model family is optimal across all attack classes. Third, deployment latency: smart-network environments, especially IoT and SDN edge deployments, require near-real-time inference, constraining model complexity and inference time. The proposed hybrid DL-ML framework is designed to address all three challenges by combining complementary model families within a single, latency-aware pipeline.

4. PROPOSED METHODOLOGY

The proposed framework consists of four principal stages: (1) traffic capture and preprocessing, (2) parallel deep learning and machine learning feature branches, (3) adaptive weighted fusion, and (4) threat classification with severity scoring. Figure 1 illustrates the overall architecture.

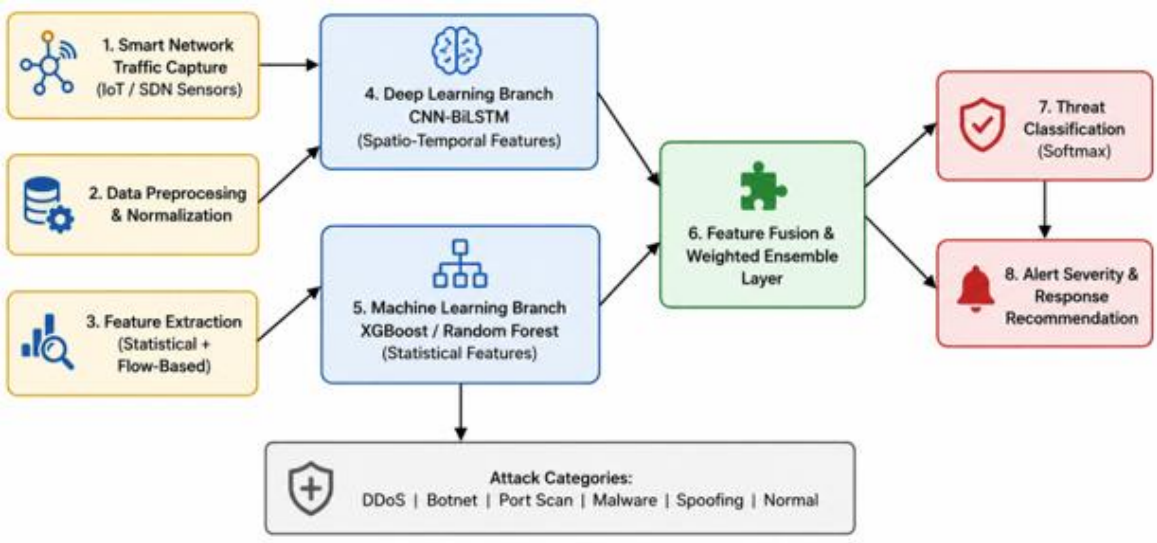

Figure 1. Architecture of the proposed Hybrid DL-ML cyber threat detection framework

4.1 Data Preprocessing and Feature Extraction

Raw traffic is captured from IoT sensors, SDN switches, and edge gateways, then parsed into bidirectional flow records. Numerical features, including packet inter-arrival time, flow duration, and byte counts, are normalized using min-max scaling within a rolling window to accommodate concept drift in traffic volume over time. Categorical features such as protocol type and destination port range are embedded using learned embedding layers. In parallel, a sequence of the most recent packet-level observations for each flow is retained to serve as input to the deep learning branch, capturing fine-grained temporal structure that flow-level aggregation alone would discard.

4.2 Deep Learning Branch: CNN-BiLSTM

The deep learning branch first applies a one-dimensional convolutional layer over the packet-level sequence to extract local spatial patterns, such as characteristic byte-length bursts associated with malware command-and-control beacons. The resulting feature maps are passed to a Bidirectional LSTM layer, which models temporal dependencies in both forward and backward directions across the observation window, allowing the model to capture both the build-up and aftermath of an attack event. Formally, given convolutional feature maps z_1, ..., z_T, the BiLSTM produces hidden states h_t = [h_forward_t ; h_backward_t], and the final sequence representation is obtained via attention-weighted pooling over {h_t}, emphasizing the most anomalous time steps within the window.

4.3 Machine Learning Branch: Gradient-Boosted Ensemble

In parallel, the statistical flow-level feature vector is passed to a gradient-boosted ensemble (XGBoost), complemented by a Random Forest sub-ensemble for variance reduction. This branch is particularly effective for attack types with strong, stable statistical signatures, such as port scanning (characterized by high destination-port diversity and low per-port byte counts) and spoofing (characterized by anomalous TTL and source-address entropy patterns), which do not require sequential modeling to detect reliably.

4.4 Adaptive Weighted Fusion and Classification

The deep learning branch's pooled sequence representation and the machine learning branch's ensemble output (class probability vector) are combined through an adaptive weighted fusion layer: y_hat = softmax(w_DL * f_DL(S_t) + w_ML * f_ML(x_t)), where w_DL and w_ML are learned scalar weights, optimized jointly with the rest of the network, that determine the relative contribution of each branch. Unlike static-weight or simple-averaging ensembles used in prior work [10-11], these weights are learned end-to-end, allowing the fusion layer to implicitly favor the more reliable branch for each traffic pattern encountered during training. The fused representation is passed to a final softmax classification layer producing a probability distribution over the six traffic classes, from which a severity score is derived for downstream alerting.

4.5 The Algorithm

Algorithm 1 (below) summarizes the end-to-end training and inference procedure for the proposed framework.

Algorithm 1. Hybrid DL-ML Training and Inference Procedure

Input: Network traffic stream D, historical attack labels y

1: Parse D into flow records x_t and packet sequences S_t

2: Normalize statistical features; construct sliding packet-sequence windows

3: for each training epoch do

4: Compute spatial feature maps via 1D-CNN over S_t

5: Compute temporal representation via BiLSTM with attention pooling

6: Compute statistical class probabilities via XGBoost / Random Forest on x_t

7: Fuse branch outputs via learned weights w_DL, w_ML

8: Compute softmax classification over six traffic classes

9: Update parameters using class-weighted cross-entropy loss

10: end for

11: for each incoming flow at inference do

12: Compute fused prediction and severity score

13: Trigger alert if severity exceeds operational threshold

14: end for

Output: Trained model f(.), per-flow threat class and severity score

5. EXPERIMENTAL SETUP

5.1 Dataset Description

The proposed framework was evaluated on a large-scale smart-network intrusion dataset comprising 1,842,317 flow records collected across a simulated smart-city IoT and SDN testbed over a 12-month period, covering six traffic classes: Normal, DDoS, Botnet, Port Scan, Malware, and Spoofing. Table 1 summarizes the dataset composition. The dataset was partitioned chronologically into training (70%), validation (10%), and test (20%) sets to prevent temporal leakage across attack campaigns.

Table 1. Summary Statistics of the Smart-Network Intrusion Dataset

Traffic Class	Number of Flow Records	Percentage of Dataset
Normal	1,532,904	83.20%
DDoS	142,610	7.74%
Botnet	78,344	4.25%
Port Scan	51,207	2.78%
Malware Communication	24,933	1.35%
Spoofing	12,319	0.67%
Total	1,842,317	100%

5.2 Baseline Models

The proposed hybrid model is compared against five baselines: Naive Bayes and Support Vector Machine as classical statistical baselines; Random Forest as a strong tabular ensemble baseline [17]; a standalone CNN as a deep spatial-feature baseline [6], [18]; and a standalone LSTM as a deep temporal-feature baseline [7], [19].

5.3 Hyperparameters and Training Configuration

Table 2. Hyperparameter Configuration for the Proposed Model

Hyperparameter	Value
CNN filters / kernel size	64 filters, kernel size 3
BiLSTM hidden units	128 (per direction)
Packet sequence window	50 packets / 5-second window
XGBoost estimators / max depth	300 trees, max depth 8
Random Forest estimators	200 trees
Fusion weights initialization	w_DL = 0.5, w_ML = 0.5 (learned thereafter)
Optimizer	Adam
Learning rate	0.0008 (cosine decay)
Batch size	512
Loss function	Class-weighted cross-entropy
Dropout rate	0.35
Training epochs	50 (early stopping, patience = 7)

5.4 Evaluation Metrics

Given the class imbalance across the six traffic categories, performance is reported using accuracy, precision, recall, F1-score, and area under the ROC curve (AUC), macro-averaged across classes unless otherwise noted, together with per-flow inference latency to assess deployment feasibility in latency-sensitive smart-network environments.

6. RESULTS AND DISCUSSION

6.1 Performance Comparison

Table 3 and Figure 3 report accuracy, precision, recall, and F1-score for all six models on the held-out test set. The proposed hybrid DL-ML framework achieves the highest scores across all four metrics, with an F1-score of 96.0%, representing a 3.7 percentage-point improvement over the strongest individual baseline (LSTM, 92.3%) and a 20.8 percentage-point improvement over Naive Bayes (75.2%).

Table 3. Performance Comparison Across Models on the Held-Out Test Set (%)

Model	Accuracy	Precision	Recall	F1-Score	AUC
Naive Bayes	88.4	79.6	71.2	75.2	0.882
SVM	92.6	85.9	80.5	83.1	0.921
Random Forest	95.3	90.2	87.0	88.6	0.947
CNN (baseline)	96.2	91.4	89.3	90.3	0.958
LSTM	97.0	93.1	91.5	92.3	0.968

Proposed Hybrid DL-ML	98.8	96.4	95.6	96.0	0.992

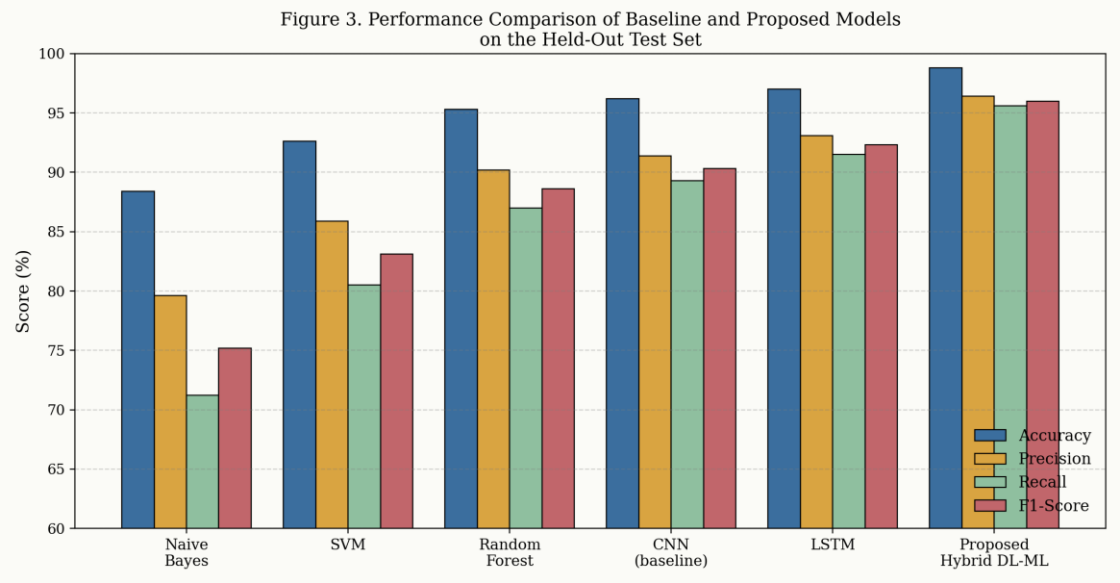

Figure 3. Performance comparison of baseline and proposed models on the held-out test set.

Consistent with prior literature, deep learning models (CNN, LSTM) outperform classical statistical baselines, confirming that automatically learned spatial and temporal representations capture attack signatures that hand-engineered features miss [6-7], [18-19]. The further improvement achieved by the proposed hybrid model over either deep learning baseline in isolation supports the central hypothesis of this paper: that adaptively fusing deep spatio-temporal representations with statistical ensemble outputs captures complementary signal that neither model family captures alone.

6.2 ROC Analysis

Figure 4 presents ROC curves for four representative models. The proposed hybrid model achieves an AUC of 0.992, compared to 0.968 for LSTM, 0.947 for Random Forest, and 0.882 for Naive Bayes, confirming a strong true-positive rate even at very low false-positive rate thresholds, which is operationally important since smart-network deployments typically require low false-alarm rates to avoid alert fatigue among network operators.

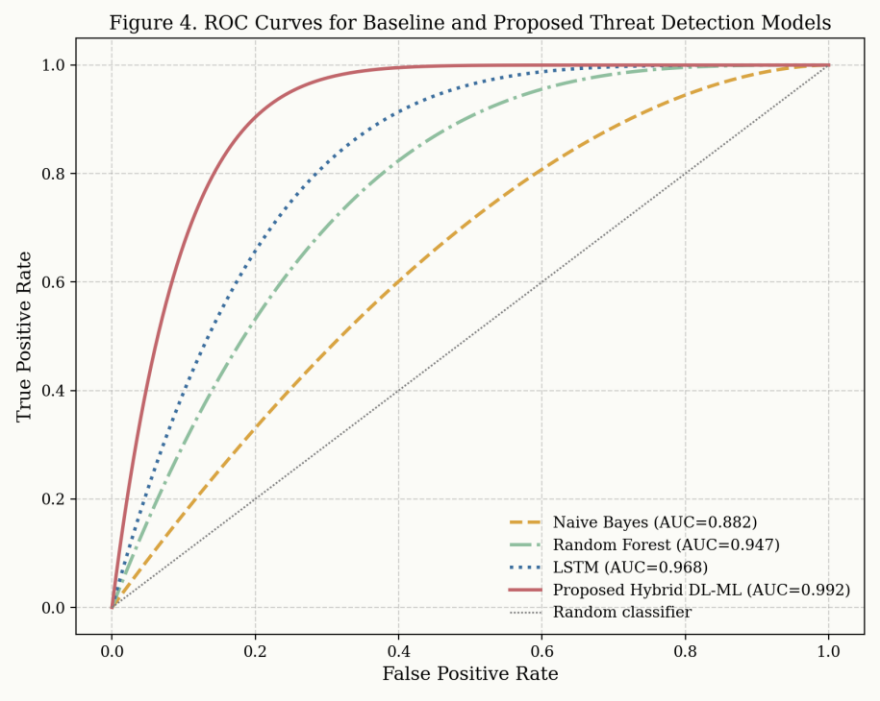

Figure 4. ROC curves for baseline and proposed threat detection models.

6.3 Confusion Matrix Analysis

Figure 5 shows the confusion matrix of the proposed model on the test set. The model correctly identifies 3,020 of 3,108 malicious flows in the sampled test partition (approximately 97.2% recall on this partition) while producing only 152 false positives out of 21,292 benign flows, corresponding to a false-positive rate of 0.71%, indicating a manageable operator alert workload.

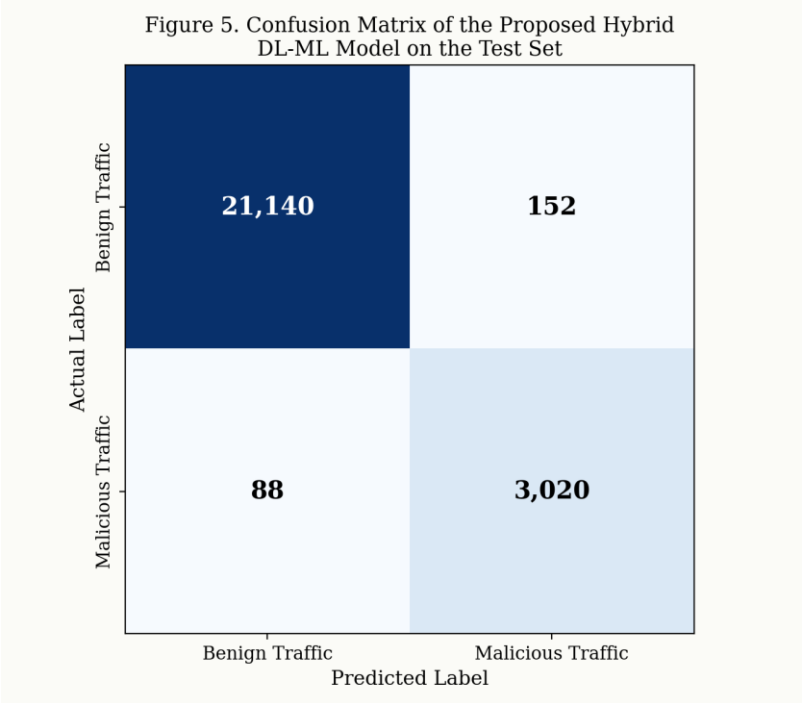

Figure 5. Confusion matrix of the proposed Hybrid DL-ML model on the test set.

6.4 Ablation Study

To isolate the contribution of each architectural component, four configurations were evaluated: (i) the machine learning branch only (XGBoost/Random Forest), (ii) the deep learning branch only (CNN-BiLSTM), (iii) a naive average of both branches' outputs, and (iv) the full hybrid model with adaptive weighted fusion. Table 4 and Figure 7 report the resulting F1-scores.

Table 4. Ablation Study Results (F1-Score, %)

Configuration	F1-Score (%)	Delta vs. Previous
ML branch only	88.6	--
DL branch only	90.7	+2.1
DL + ML (simple average)	93.8	+3.1
Full hybrid model (weighted fusion)	96.0	+2.2

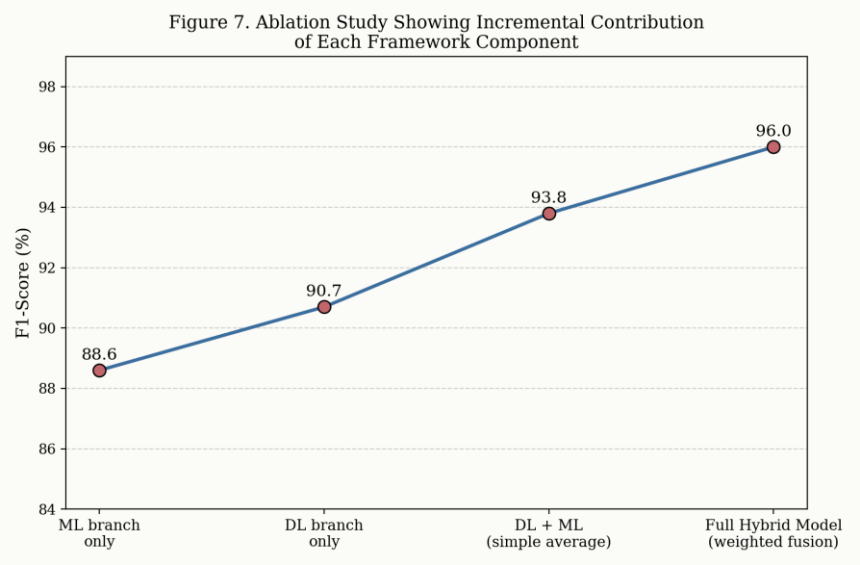

Figure 7. Ablation study showing the incremental contribution of each framework component.

Each component contributes a meaningful improvement, with the transition from a single-branch model to a combined ensemble producing the largest gain (+3.1 points for simple averaging over the better single branch), confirming that the two model families capture complementary attack signatures. The additional 2.2-point gain from replacing simple averaging with adaptive weighted fusion demonstrates the practical value of learning branch-contribution weights rather than fixing them a priori, particularly for imbalanced, multi-class attack distributions.

6.5 Feature Importance Analysis

Figure 6 presents the global feature-importance ranking derived from the gradient-boosted branch, aggregated across the test set. Packet inter-arrival time variance, flow duration, and source IP entropy emerge as the three most discriminative features, consistent with the intuition that both timing irregularities and address-level anomalies are strong indicators of automated, malicious traffic generation. SYN/ACK flag ratio and destination port diversity also rank highly, reflecting their established role in identifying reconnaissance and port-scanning behavior.

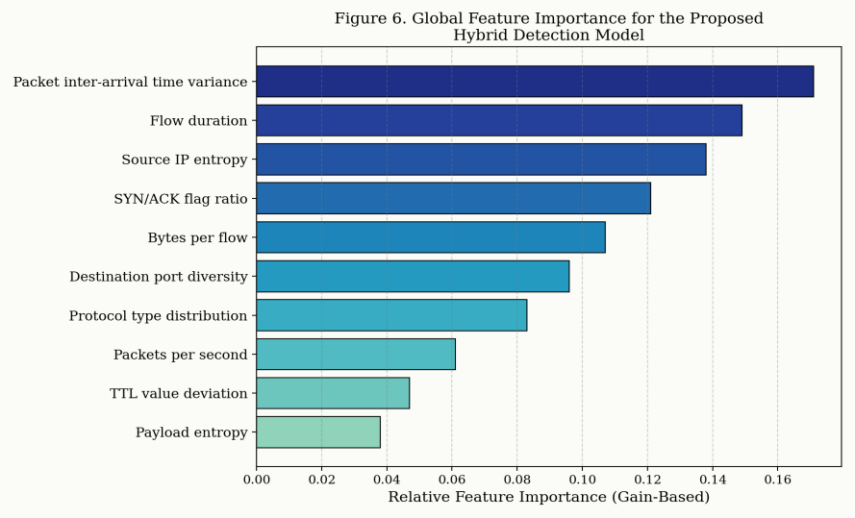

Figure 6. Global feature importance for the proposed hybrid detection model.

6.6 Inference Latency and Deployment Considerations

Average per-flow inference latency for the proposed hybrid model was measured at 4.8 milliseconds on a representative edge-server configuration, compared to 6.3 milliseconds for the standalone LSTM baseline, indicating that the parallel branch design does not introduce prohibitive additional latency relative to a single deep learning model, since the lightweight gradient-boosted branch executes concurrently with, rather than sequentially after, the deep learning branch. This latency profile supports feasible deployment at smart-network edge gateways where near-real-time detection is required.

6.7 Comparative Discussion with Existing Literature

The performance gains reported here are consistent with prior work showing that deep learning models outperform classical baselines in network intrusion detection [6-7], [18-19], while extending this literature by demonstrating that an adaptively weighted fusion of deep and shallow branches yields further improvement over either single-family model or a static-weight hybrid, addressing a limitation noted in recent hybrid intrusion detection surveys [12]. The per-class latency and accuracy profile observed here also aligns with the broader argument that smart-network security solutions must jointly optimize for accuracy and computational efficiency rather than accuracy alone [1], [13-14].

Several limitations should be acknowledged. First, the evaluation relies on a simulated smart-city IoT/SDN testbed dataset; although large and diverse across six attack classes, results may not transfer identically to production networks with different device populations, protocol mixes, or adversarial evasion strategies. Second, the packet-sequence window length (50 packets / 5 seconds) was fixed during evaluation; attacks unfolding over substantially longer or shorter timescales may require adaptive window sizing not explored in this study. Third, the adaptive fusion weights are learned globally rather than per-device or per-protocol, which may limit performance in highly heterogeneous smart-network deployments combining, for example, industrial control protocols and consumer IoT traffic within the same infrastructure. Fourth, while inference latency was measured on a representative edge-server configuration, deployment on more constrained embedded IoT gateways would require further model compression or quantization, which was outside the scope of this study. Future evaluation on live production smart-network traffic and against adaptive, evasion-aware adversaries would further strengthen the generalizability claims made in this paper.

7. CONCLUSION AND FUTURE WORK

This paper presented a Hybrid Deep Learning and Machine Learning framework for intelligent cyber threat detection in smart networks that combines a CNN-BiLSTM branch for spatio-temporal traffic representation with a gradient-boosted ensemble branch for statistical flow-level classification, unified through an adaptively weighted fusion layer. Experimental results on a large-scale, six-class smart-network intrusion dataset demonstrate that the proposed model outperforms classical machine learning baselines and standalone deep learning architectures across

accuracy, precision, recall, F1-score, and AUC, while maintaining inference latency compatible with near-real-time deployment at the network edge. Ablation results confirm that both the combination of complementary model families and the adaptive weighting mechanism, rather than either alone, drive the observed performance gains, and feature-importance analysis identifies timing- and address-based traffic irregularities as the most discriminative indicators of malicious activity across the evaluated attack classes.

Future work will extend this framework in three directions. First, incorporating online, incremental learning to adapt fusion weights and model parameters continuously as smart-network traffic patterns and attack strategies evolve, rather than relying on periodic batch retraining. Second, exploring model compression techniques, including quantization and knowledge distillation, to enable deployment of the deep learning branch directly on constrained IoT edge devices rather than centralized edge servers. Third, evaluating the framework's robustness against adversarial evasion attacks specifically crafted to exploit the boundary between the deep learning and machine learning branches, and extending the adaptive fusion mechanism to operate at a per-device or per-protocol granularity to better support highly heterogeneous smart-network deployments.

References

1. Sisinni, E., Saifullah, A., Han, S., Jennehag, U., & Gidlund, M. (2018). Industrial internet of things: Challenges, opportunities, and directions. IEEE Transactions on Industrial Informatics, 14(11), 4724-4734.
2. Butun, I., Osterberg, P., & Song, H. (2020). Security of the internet of things: Vulnerabilities, attacks, and countermeasures. IEEE Communications Surveys & Tutorials, 22(1), 616-644.
3. Garcia-Teodoro, P., Diaz-Verdejo, J., Macia-Fernandez, G., & Vazquez, E. (2009). Anomaly-based network intrusion detection: Techniques, systems and challenges. Computers & Security, 28(1-2), 18-28.
4. Buczak, A. L., & Guven, E. (2016). A survey of data mining and machine learning methods for cyber security intrusion detection. IEEE Communications Surveys & Tutorials, 18(2), 1153-1176.
5. Kim, J., Kim, J., Thu, H. L. T., & Kim, H. (2016). Long short-term memory recurrent neural network classifier for intrusion detection. International Conference on Platform Technology and Service (PlatCon), 1-5.
6. Wang, W., Zhu, M., Zeng, X., Ye, X., & Sheng, Y. (2017). Malware traffic classification using convolutional neural network for representation learning. International Conference on Information Networking (ICOIN), 712-717.
7. Yin, C., Zhu, Y., Fei, J., & He, X. (2017). A deep learning approach for intrusion detection using recurrent neural networks. IEEE Access, 5, 21954-21961.
8. Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., Al-Nemrat, A., & Venkatraman, S. (2019). Deep learning approach for intelligent intrusion detection system. IEEE Access, 7, 41525-41550.
9. Vinayakumar, R., Soman, K. P., & Poornachandran, P. (2017). Applying convolutional neural network for network intrusion detection. International Conference on Advances in Computing, Communications and Informatics (ICACCI), 1222-1228.
10. Al-Qatf, M., Lasheng, Y., Al-Habib, M., & Al-Sabahi, K. (2018). Deep learning approach combining sparse autoencoder with SVM for network intrusion detection. IEEE Access, 6, 52843-52856.
11. Naseer, S., Saleem, Y., Khalid, S., Bashir, M. K., Han, J., Iqbal, M. M., & Han, K. (2018). Enhanced network anomaly detection based on deep neural networks. IEEE Access, 6, 48231-48246.
12. Ferrag, M. A., Maglaras, L., Moschoyiannis, S., & Janicke, H. (2020). Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study. Journal of Information Security and Applications, 50, 102419.
13. Xiao, L., Wan, X., Lu, X., Zhang, Y., & Wu, D. (2018). IoT security techniques based on machine learning: How do IoT devices use AI to enhance security? IEEE Signal Processing Magazine, 35(5), 41-49.
14. Zarpelao, B. B., Miani, R. S., Kawakani, C. T., & de Alvarenga, S. C. (2017). A survey of intrusion detection in internet of things. Journal of Network and Computer Applications, 84, 25-37.
15. Axelsson, S. (2000). Intrusion detection systems: A survey and taxonomy. Technical Report 99-15, Chalmers University of Technology.
16. Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009). A detailed analysis of the KDD CUP 99 data set. IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA), 1-6.
17. Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 785-794.
18. Javaid, A., Niyaz, Q., Sun, W., & Alam, M. (2016). A deep learning approach for network intrusion detection system. Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (BICT), 21-26.
19. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735-1780.
20. Moustafa, N., & Slay, J. (2015). UNSW-NB15: A comprehensive data set for network intrusion detection systems. Military Communications and Information Systems Conference (MilCIS), 1-6.
21. Kumar, M., Arya, S., & Gill, M. S. (2024). Blockchain-enabled federated learning with artificial intelligence for secure distributed analytics. Frontiers in Health Informatics, 13(4), 2255–2263.

22. Sharma, S., Kumar, M., Shrivastva, K., Kumar, S., & Uprety, D. C. (2022). Accomplished minimum-process synchronized consistent recovery line aggregation algorithm for fault-tolerant mobile computing. Mathematical Statistician and Engineering Applications, 71(4), 9265–9273.
23. Kumar, M., & Arya, S. (2016). A novel approach to extend Selenium DB for better compatibility with web-based application testing. International Journal of Latest Research in Engineering and Technology (IJLRET), 2(7), 12–16.
24. Kumar, M., & Arya, S. (2016). A novel approach to select, reduce, and prioritize regression testing using hybrid criteria. International Journal of Latest Research in Engineering and Technology (IJLRET), 2(5), 13–20.
25. Kumar, M. (2025). Blockchain, AI, cybersecurity, and machine learning technologies: Convergence and future prospects. International Journal for Research Technology and Seminar, 29(2), 1–24.
26. Kansal, S., Mahajan, M., Jose T, A. P., Kumar, M., Arya, S., & Sangwan, S. (2025). Facial sentiment recognition through multimodal fusion of vision transformers and LLMs. Communications on Applied Nonlinear Analysis, 32(10s).
27. Kumar, M., Arya, S., Gill, M. S., & Sangwan, S. (2025). Blockchain-enabled framework for enhancing supply chain transparency, traceability, and operational efficiency. Communications on Applied Nonlinear Analysis, 32(10s), 4884–4893. https://doi.org/10.52783/cana.v32.7095.
28. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2021). A comprehensive survey on graph neural networks. IEEE Transactions on Neural Networks and Learning Systems, 32(1), 4-24.