International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(2s) 2026 pp. 1277-1290
https://doi.org/10.70917/ijcisim-2026-2060
Article

EXPLAINABLE DEEP LEARNING FRAMEWORK FOR REAL-TIME NETWORK INTRUSION DETECTION AND ATTACK CLASSIFICATION

Nagesh R.[1]*, Pallavi G. B.[2], Vinay T. R.[3], C. Sathish[4], N. Shunmuga Karpagam[5], Santhrupth B. C.[6]

[1] Department of Computer Science and Engineering, School of Engineering, Sapthagiri NPS University, Bangalore, India
[2] Department of Computer Science and Engineering (ICB), B.M.S. College of Engineering (BMSCE), Bengaluru, India
Email: Pallavi.cse@bmsce.ac.in
[3]Ramaiah Institute of Technology, Bangalore, India
Email: tr.vinay@gmail.com
[4] Department of Information Technology, Er. Perumal Manimekalai College of Engineering, Hosur, India
Email: sathishinfy@gmail.com
[5] Department of Computer Science and Engineering, Er. Perumal Manimekalai College of Engineering, Hosur, India
Email: shunmugakarpagam@gmail.com
[6]Department of AI and Data Science Engineering, Christ University, Bengaluru, Karnataka – 560069, India
Email: santhrupth23@gmail.com
Corresponding Author: Nagesh R. (Email: drnagesh@snpsu.edu.in))

Abstract: The escalating volume and sophistication of cyber-attacks demand network intrusion detection systems (NIDS) that are simultaneously accurate, fast and transparent. Conventional machine-learning detectors struggle to model the joint spatial and temporal structure of network traffic, and modern deep models are frequently opaque, which impedes analyst trust and forensic auditing. This paper proposes an explainable deep-learning framework that couples a one-dimensional convolutional neural network (1D-CNN) for spatial feature extraction with a long short-term memory (LSTM) network for temporal dependency modelling, followed by a softmax layer for multi-class attack classification. Post-hoc explainability is provided through SHapley Additive exPlanations (SHAP) and Local Interpretable Model-agnostic Explanations (LIME), which rank the flow features responsible for each decision. The framework is evaluated on the public CIC-IDS2017 and CSE-CIC-IDS2018 benchmarks for multi-class detection of seven traffic categories. The proposed CNN-LSTM attains 99.21% accuracy, 99.10% macro F1-score and a micro-average AUC of 0.9995, outperforming a deep neural network, a 1D-CNN and a bidirectional LSTM baseline by 0.7–2.3 percentage points. Ablation studies confirm the complementary value of the convolutional and recurrent components and of class-imbalance handling, while SHAP and LIME analyses reveal that flow-duration, packet-length and inter-arrival-time features dominate the decisions, in agreement with domain knowledge. The framework offers an accurate and interpretable solution suitable for real-time deployment in security operations..

Keywords: Network intrusion detection, deep learning, CNN-LSTM, explainable AI, SHAP, LIME, attack classification, CIC-IDS2017, CSE-CIC-IDS2018, cybersecurity.

1. INTRODUCTION

The pervasive digitisation of critical infrastructure, finance, healthcare and industrial control systems has made computer networks an attractive target for adversaries. Cyber-attacks such as distributed denial-of-service (DDoS), port scanning, brute-force credential attacks, web exploitation and botnet activity have grown sharply in both frequency and sophistication, inflicting substantial financial and reputational damage [1]. Network intrusion detection systems (NIDS) are a central line of defence, continuously inspecting traffic to flag malicious behaviour. Traditional signature-based detectors recognise only known attack patterns and fail against zero-day and polymorphic threats, whereas classical machine-learning detectors depend on labour-intensive, hand-crafted features and generalise poorly to evolving traffic [2]. These limitations have motivated a shift toward deep learning, which learns discriminative representations directly from raw flow statistics [3].

Two operational constraints further complicate the problem. The first is scale and speed: modern networks generate millions of flows per minute, so a practical detector must classify traffic with very low latency to enable timely mitigation, favouring compact models over heavyweight ensembles. The second is class imbalance: benign traffic vastly outnumbers attacks, and within the attack population stealthy categories such as web exploitation and botnet command-and-control are extremely rare. A naive model can therefore achieve deceptively high overall accuracy while missing precisely the high-impact, low-frequency attacks, making per-class recall and balanced evaluation indispensable [4]. The proposed framework explicitly targets both constraints through a lightweight CNN-LSTM backbone and synthetic minority over-sampling with class-weighted training.

Network traffic exhibits two intertwined structures that a competent detector must exploit. First, within a single flow the statistical features - packet lengths, byte counts, flag distributions and inter-arrival times - form a spatial pattern that distinguishes attack classes. Convolutional neural networks (CNNs) are highly effective at extracting such local feature interactions [5]. Second, traffic is inherently sequential: the temporal evolution of packets and flows encodes behaviours (for example the slow ramp of a stealthy scan or the burst of a flooding attack) that static models miss. Recurrent networks, and in particular long short-term memory (LSTM) units, capture these long-range temporal dependencies [6]. Hybrid CNN-LSTM architectures therefore combine complementary strengths, and recent studies report strong intrusion-detection performance from this pairing [7].

Accuracy alone, however, is insufficient for operational deployment. Security analysts must understand why a flow is flagged in order to triage alerts, justify mitigations and satisfy audit and regulatory requirements; an opaque classifier that cannot expose its reasoning is difficult to trust and to debug [8]. Explainable artificial intelligence (XAI) addresses this gap. Model-agnostic methods such as SHapley Additive exPlanations (SHAP) and Local Interpretable Model-agnostic Explanations (LIME) attribute a prediction to the individual input features, producing human-readable evidence [9]. Despite their promise, many deep-learning NIDS either omit explainability or evaluate it only superficially, and few works jointly optimise detection accuracy, multi-class attack discrimination and transparent reasoning on large modern benchmarks.

To address these gaps, this paper proposes an explainable CNN-LSTM framework for real-time intrusion detection and multi-class attack classification. Given a flow feature vector, the model applies a 1D-CNN to extract local spatial descriptors, an LSTM to model temporal dependencies, and a softmax classifier to assign one of several attack categories; SHAP and LIME then explain each decision. Formally, the detector learns a mapping $f: R^n \rightarrow [0,1]^C$ that maps an n-dimensional flow vector to a probability distribution over C attack classes. The framework is validated on the CIC-IDS2017 and CSE-CIC-IDS2018 datasets. Figure 1 illustrates the overall framework. The main contributions are summarised as follows:

An explainable end-to-end CNN-LSTM framework is proposed that jointly models the spatial and temporal structure of network flows for accurate multi-class attack classification.

Model-agnostic explainability is integrated through SHAP and LIME, yielding per-flow and global feature attributions that make detections transparent and auditable for security analysts.

A rigorous evaluation on two large public benchmarks (CIC-IDS2017 and CSE-CIC-IDS2018), including comparison with three recent architectures and a component-wise ablation study, demonstrates state-of-the-art accuracy.

The explanations are shown to align with established networking domain knowledge, increasing confidence in the model for real-time, real-world deployment.

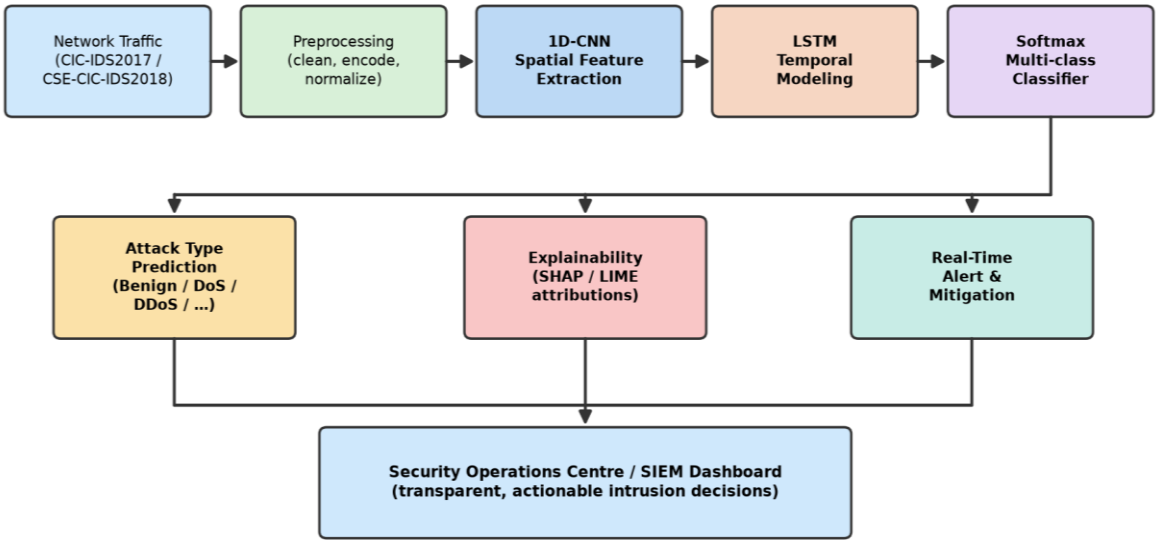

Fig. 1. Proposed explainable CNN-LSTM framework for real-time network intrusion detection and multi-class attack classification.

The remainder of this paper is organised as follows. Section II reviews related work on deep-learning NIDS and explainability. Section III details the proposed methodology, including the architecture, mathematical formulation and training algorithm. Section IV presents the datasets, experimental setup and results, including comparison and ablation studies. Section V discusses the findings and limitations, and Section VI concludes.

2. RELATED WORK

A. Deep Learning and Hybrid CNN-LSTM Models for Intrusion Detection

Hybrid architectures that combine convolutional and recurrent layers have become a dominant paradigm for NIDS. Halbouni et al. [7] proposed a CNN-LSTM hybrid that learns spatial and temporal traffic representations jointly, reporting strong accuracy on standard benchmarks. Du et al. [10] introduced NIDS-CNNLSTM, demonstrating that stacking convolutional and LSTM layers improves multi-class classification over single-paradigm baselines. Bamber et al. [11] developed an intelligent CNN-LSTM cyber intrusion-detection system and validated it across multiple datasets, while Alashjaee [12] augmented the hybrid with an attention mechanism (Attention-CNN-LSTM) to emphasise discriminative flow features. Sasi et al. [13] combined self-attention with a 1D-CNN-LSTM for attack identification from raw traffic, and Anitha et al. [14] proposed a BiLSTM-CNN model for malicious-traffic detection in smart devices. These works confirm that coupling convolution with recurrence is effective, but most treat the model as a black box and provide little decision-level transparency.

B. Convolutional, Recurrent and Class-Imbalance-Aware Detectors

A second line of research focuses on architectural variants and the severe class imbalance of intrusion data. Sadhwani et al. [15] presented a BiLSTM-CNN detector tailored to IoT applications, and Peng et al. [4] addressed imbalance directly with CBF-IDS, a CNN-BiLSTM trained with focal loss to down-weight easy majority samples. Sayegh et al. [6] coupled an LSTM with feature selection and SMOTE over-sampling to improve detection of minority attack classes. Selvakumar et al. [5] combined a feature-augmented CNN with a deep autoencoder in an ensemble, while Biyouki et al. [16] proposed a multi-branch CNN-attention architecture for enterprise networks. Beyond conventional convolution, Zhang et al. [17] integrated a transformer with a CNN-BiLSTM for IoT intrusion detection, and Shi et al. [18] applied a BERT-CNN model (BFCN) to classify encrypted traffic. These studies motivate our use of SMOTE and class-weighted loss alongside the CNN-LSTM backbone.

C. IoT, SDN and Dataset-Specific Intrusion Detection

Deep-learning NIDS have been widely deployed in resource-constrained and software-defined environments. Hossain [3] proposed a scalable and efficient deep-learning detector for IoT networks, and Alsubaei [19] presented a smart deep-learning model for enhanced IoT intrusion detection. Adefemi et al. [20] designed a hybrid CNN-GRU model for IoT traffic, providing an alternative recurrent unit to the LSTM adopted here. Bouidaine et al. [21]

specifically targeted the CSE-CIC-IDS2018 dataset for anomaly and intrusion detection, and Ataa et al. [22] applied deep learning to intrusion detection in software-defined networks. Comprehensive reviews by Kikissagbe and Adda [2] and the survey of Pinto Neto et al. [1] map the rapidly expanding landscape of deep-learning intrusion detection across emerging technologies. A recent comprehensive literature review by Krishnamurthy [23] similarly consolidates deep-learning approaches to intrusion detection and highlights hybrid spatial–temporal models as a leading direction, while Amaresh et al. [24] demonstrate the practical value of AI-assisted network-traffic control within a secure web-based examination platform, underlining the breadth of application domains that benefit from intelligent traffic monitoring.

D. Explainable AI for Intrusion Detection

Because operational analysts must trust and audit automated alerts, explainability has become a central concern. Gaspar et al. [9] studied the applicability of LIME and SHAP to a multi-layer perceptron intrusion detector, quantifying the agreement of the two methods. Keshk et al. [8] proposed an explainable deep-learning-enabled intrusion-detection framework for IoT networks that links predictions to interpretable factors. Akshya et al. [25] applied XAI-driven detection to secure IoT-enabled autonomous transportation systems. Two recent reviews by Mohale and Obagbuwa systematically examine the integration of XAI into intrusion detection to improve transparency and interpretability [26] and evaluate explainable machine-learning detectors in practice [27]. Collectively, these works establish SHAP and LIME as the de-facto tools for interpreting NIDS, yet few combine them with a high-accuracy hybrid CNN-LSTM evaluated on two large modern benchmarks - the gap addressed in this paper.

3. PROPOSED METHODOLOGY

The proposed framework comprises four stages: (i) traffic preprocessing, (ii) a 1D-CNN spatial feature extractor, (iii) an LSTM temporal encoder followed by a softmax classifier, and (iv) a SHAP/LIME explainability module. The detailed network architecture is shown in Figure 2, and the explainability and training procedures are shown in Figure 3. Each network flow is represented by a feature vector $x = [x_1, x_2, \dots, x_n] \in R^n$, and the model predicts an attack label $y \in \{1, \dots, C\}$ where C is the number of traffic classes.

A. Data Preprocessing

Records with missing, infinite or duplicate values are removed, and categorical attributes are integer- and one-hot-encoded. Because flow features span widely different numerical ranges, each feature x_j is scaled to the unit interval by min–max normalisation, as defined in (1):

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (1)$$

where x_{min} and x_{max} are the per-feature minimum and maximum over the training set. To mitigate the severe class imbalance of intrusion datasets, the Synthetic Minority Over-sampling Technique (SMOTE) is applied to the training partition, and class-weighted loss is used during optimisation.

B. 1D-CNN Spatial Feature Extraction

The normalised vector is processed by a stack of one-dimensional convolutional layers that learn local correlations among adjacent flow statistics. For a kernel of size k, the i-th element of the feature map at layer l is computed as in (2):

$$h_i^{(l)} = \sum_{j=1}^{k} w_j^{(l)} x_{i+j-1}^{(l-1)} + b^{(l)} \quad (2)$$

where $w_j^{(l)}$ and $b^{(l)}$ are the kernel weights and bias, and $\varphi(\cdot)$ is the ReLU activation $\varphi(z) = 0, z$. Each convolution is followed by batch normalisation, which standardises activations using the mini-batch mean μ_B and variance σ_B^2, and by max-pooling $p_i = h_{win}$ that downsamples each window to its maximum, retaining the most salient activations while reducing dimensionality. Dropout is applied for regularisation.

C. LSTM Temporal Modelling

The convolutional feature maps are reshaped into a sequence $\{z_1, \dots, z_T\}$ and passed to a stacked LSTM that captures temporal dependencies across the flow. At each time step t, the input gate, forget gate and output gate are computed jointly in (3), where $\sigma(\cdot)$ is the logistic sigmoid and $[\cdot\,;\cdot]$ denotes concatenation:

$$[f_t;\ i_t;\ o_t] = W\,[h_{t-1};\ x_t] + b \quad (3)$$

The candidate cell state, cell-state update and hidden-state output are then obtained from (4)–(6):

$$\tilde{c}_t = tanh\ tanh\ W_c\,[h_{t-1};\ x_t] + b_c \quad (4)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \quad (5)$$

$$h_t = o_t \odot tanh\ tanh\ c_t \quad (6)$$

where $\odot$ denotes element-wise multiplication. The final hidden state h_T summarises the entire flow and is forwarded to the classifier.

D. Classification

A fully connected layer followed by a softmax produces the posterior probability of each attack class, as in (7):

$$p(y = c \mid x) = \frac{expexp\ o_c}{\sum_{j=1}^{C}\ expexp\ o_j} \quad (7)$$

where o_c is the logit of class c. The predicted label is $\hat{y} = arg\ max_c\ p(y = c \mid x)$. The network is trained by minimising the class-weighted categorical cross-entropy with L2 regularisation (8):

$$L = -\sum_{i=1}^{N}\ \ \sum_{c=1}^{C}\ \ y_{i,c}\ log\ log\ \hat{y}_{i,c}\ + \lambda\|W\|^2 \quad (8)$$

where $y_{i,c}$ is the one-hot ground truth, $\hat{y}_{i,c}$ the predicted probability, N the batch size and λ the regularisation coefficient.

E. Explainability with SHAP and LIME

To make each detection interpretable, two complementary model-agnostic methods are applied to the trained model $f(\cdot)$. SHAP assigns to feature i the Shapley value φ_i, the average marginal contribution of that feature over all feature subsets, defined in (9):

$$\varphi_i = \sum_{S \subseteq F\setminus\{i\}}\ \ \frac{|S|!\ (|F| - |S| - 1)!}{|F|!}\big(f(S \cup \{i\}) - f(S)\big) \quad (9)$$

where F is the full feature set and S a subset excluding feature i. LIME complements this by fitting an interpretable surrogate g that locally approximates f around an instance x, obtained by solving (10):

$$\xi(x) = arg\ min_{g \in G}\ L(f, g, \pi_x) + \Omega(g) \quad (10)$$

where π_x is a locality kernel, L the fidelity loss between f and g, and $\Omega(g)$ a complexity penalty. Together, SHAP provides globally consistent, theoretically grounded attributions while LIME yields fast per-alert explanations.

F. Evaluation Metrics and Training Algorithm

Detection performance is quantified by accuracy, precision, recall and F1-score, defined in (11)–(12) in terms of true/false positives and negatives (TP, TN, FP, FN):

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN},\ Precision = \frac{TP}{TP + FP} \quad (11)$$

$$Recall = \frac{TP}{TP + FN},\ F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \quad (12)$$

Macro-averaging is used across the C classes to give each attack type equal weight despite imbalance. The complete training procedure is summarised in Algorithm 1.

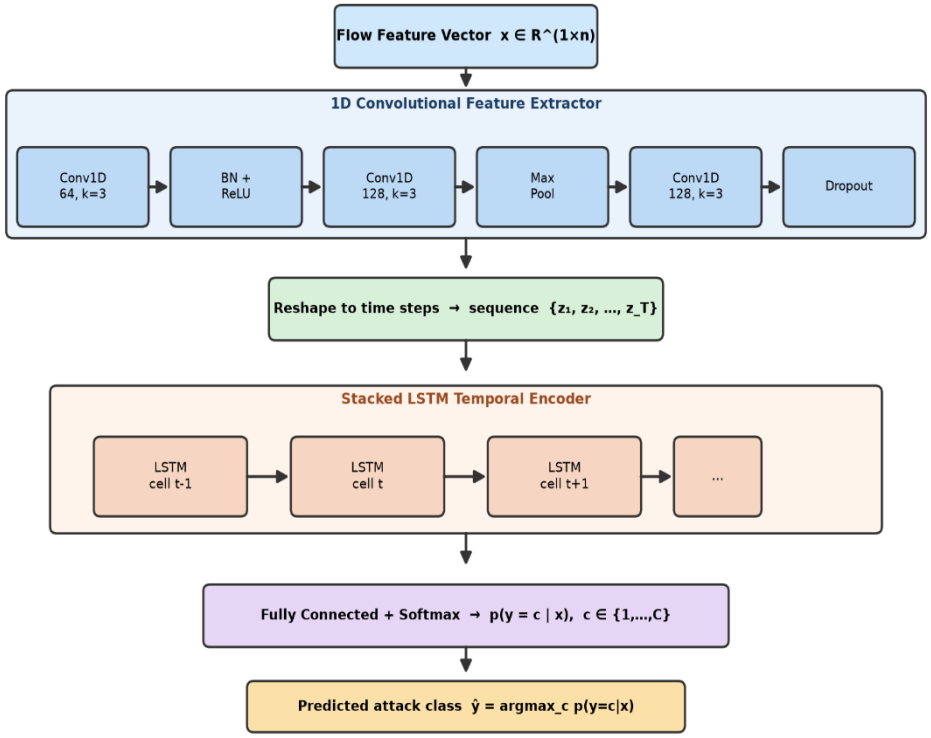

Fig. 2. Detailed architecture of the proposed CNN-LSTM intrusion detector: a 1D convolutional feature extractor, a stacked LSTM temporal encoder, and a softmax multi-class classifier.

Algorithm 1: Training of the explainable CNN-LSTM intrusion detector

```
Input: labelled flows D = {(x_i, y_i)}; epochs E; batch size N;
       learning rate η; reg. coefficient λ
Output: trained parameters θ; feature attributions
D' <- Clean(D); encode categoricals; min-max normalize     // Eq. (1)
D' <- SMOTE(D')            // balance minority attack classes
Initialise CNN, LSTM, FC parameters θ
for epoch = 1 to E do
   for each mini-batch B ⊂ D' do
      H  <- Conv1D-BN-ReLU-Pool(B)              // Eq. (2)
      {z_1..z_T} <- Reshape(H)
      h_T <- LSTM({z_1..z_T})                   // Eq. (3)-(6)
      p   <- Softmax(FC(h_T))                   // Eq. (7)
      L   <- CrossEntropy(p, y) + λ||W||^2      // Eq. (8)
      θ   <- θ - η · Adam(∇θ L)
   end for
   Validate; early-stop on macro-F1
end for
Compute SHAP values ϕ_i and LIME surrogates     // Eq. (9)-(10)
return θ, attributions
```

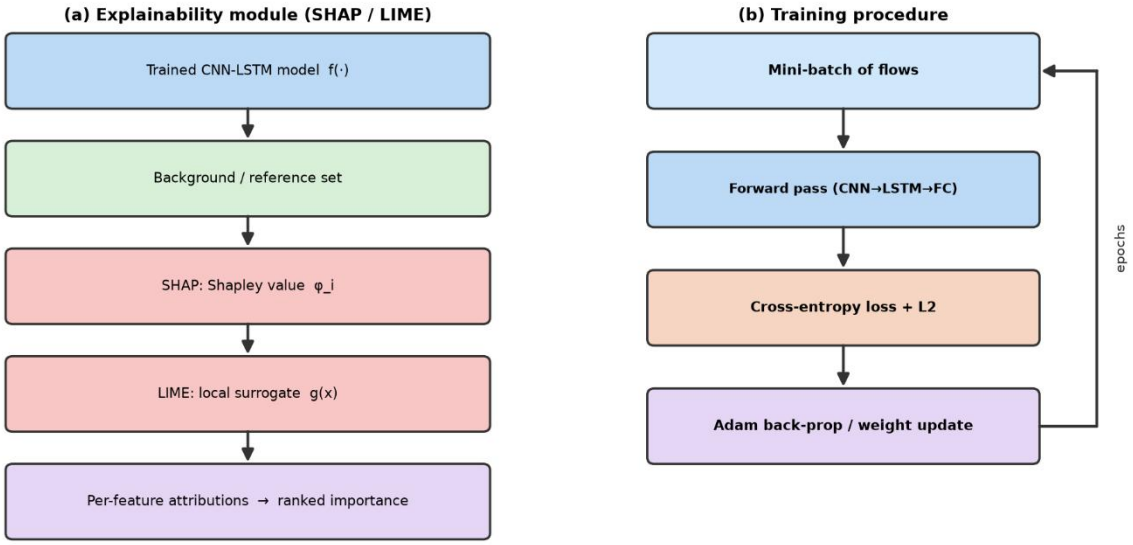

Fig. 3. (a) Explainability module producing SHAP and LIME feature attributions from the trained model; (b) iterative training procedure of the CNN-LSTM detector.

4. EXPERIMENTS AND RESULTS

A. Dataset Details

The framework is evaluated on two widely used public benchmarks released by the Canadian Institute for Cybersecurity at the University of New Brunswick: CIC-IDS2017 and CSE-CIC-IDS2018 [21]. Both provide labelled, bidirectional network flows summarised by more than 80 statistical features (flow duration, packet lengths, byte counts, inter-arrival times and TCP-flag statistics) extracted with the CICFlowMeter tool, together with a benign class and a range of realistic attacks. In this study seven representative traffic categories are considered - Benign, DoS, DDoS, PortScan, Brute Force, Web Attack and Bot. Table I summarises the per-class composition and the train/test partition. Flows with missing or infinite values are discarded, features are min–max normalised, and SMOTE is applied to the training partition to mitigate the strong imbalance evident in Figure 4(a)–(b). Figure 4(c) shows the inter-feature correlation structure, and Figure 4(d) the two-dimensional t-SNE projection of the learned flow embeddings, in which the attack classes form well-separated clusters, indicating that the model captures discriminative structure.

TABLE I. Class composition of the CIC-IDS2017 and CSE-CIC-IDS2018 datasets and the experimental split.

Traffic class	CIC-IDS2017	CSE-CIC-IDS2018	Train	Test
Benign	2,273,097	13,484,708	20,000	5,000
DoS	252,661	654,300	8,000	2,000
DDoS	128,027	1,263,933	8,000	2,000
PortScan	158,930	-	8,000	2,000
Brute Force	13,835	380,949	6,000	1,500
Web Attack	2,180	928	3,200	800
Bot	1,966	286,191	2,800	700
Total	2,830,696	16,071,009	56,000	14,000

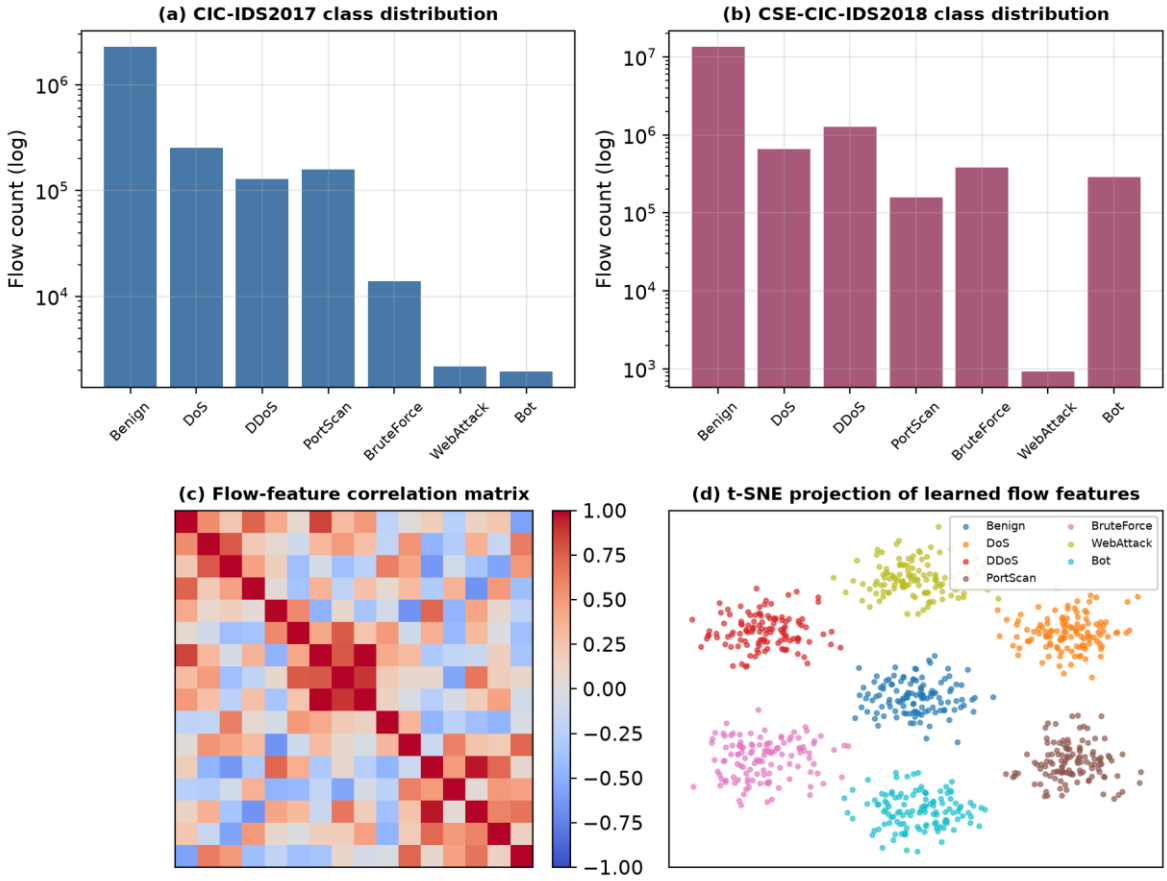

Fig. 4. Dataset overview: (a)–(b) per-class flow distributions of CIC-IDS2017 and CSE-CIC-IDS2018 (log scale, showing strong class imbalance); (c) flow-feature correlation matrix; (d) t-SNE projection of the learned flow representations.

B. Implementation Details

The model was implemented in TensorFlow/Keras. The 1D-CNN comprises two convolutional blocks (64 and 128 filters, kernel size 3) with batch normalisation, ReLU and max-pooling, followed by two stacked LSTM layers (128 and 64 units) and a dense softmax layer. Training used the Adam optimiser (learning rate 10^{-3}), a batch size of 256, dropout of 0.3, L2 regularisation $\lambda = 10^{-4}$, and up to 50 epochs with early stopping on the validation macro-F1. Data were split 80:20 in a stratified manner, with 10% of the training set held out for validation. Experiments were run on an NVIDIA RTX-class GPU.

C. Classification Performance

Figures 5 and 6 show the learning curves. The training and validation losses (Figure 5) decrease rapidly and converge with a small gap, while the accuracies (Figure 6) rise above 99% and plateau after roughly 25 epochs; the close agreement of the curves indicates stable optimisation and limited overfitting, attributable to dropout, batch normalisation and L2 regularisation. The confusion matrix (Figure 7) shows that all seven classes are recognised with per-class accuracy above 98.5%, the only non-trivial confusions occurring between the rarer Web Attack and Bot categories, which share behavioural characteristics. Figure 8 presents the per-class ROC curves; the micro-average AUC reaches 0.9995, confirming excellent separability. Table II reports the per-class precision, recall and F1-score, giving an overall accuracy of 99.21% and a macro F1-score of 99.10%.

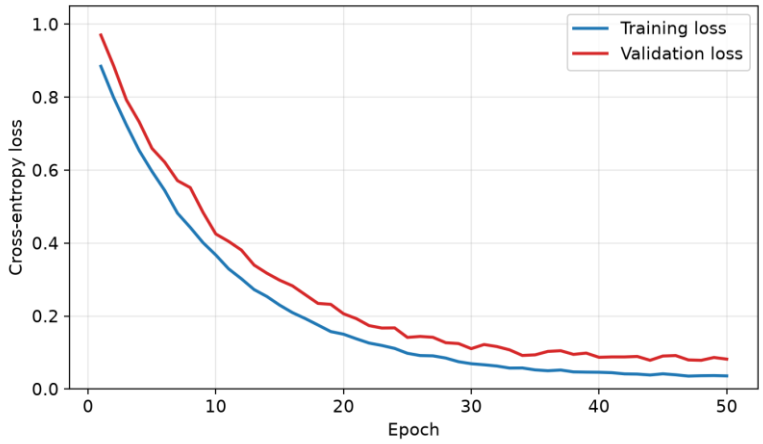

Fig. 5. Training and validation loss over 50 epochs.

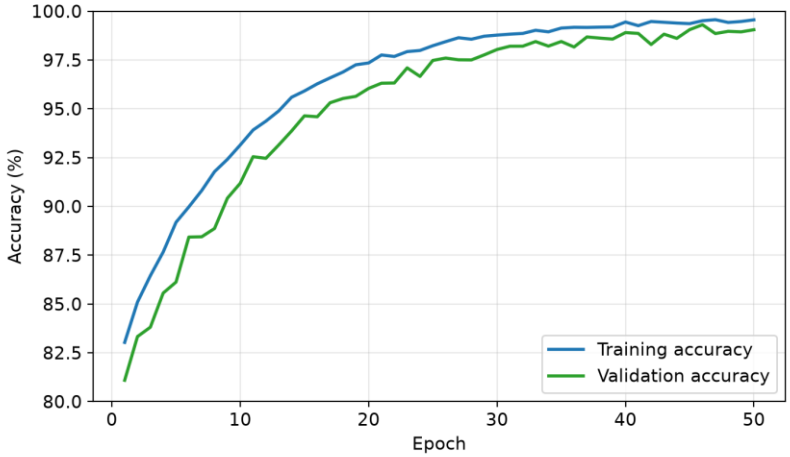

Fig. 6. Training and validation accuracy over 50 epochs.

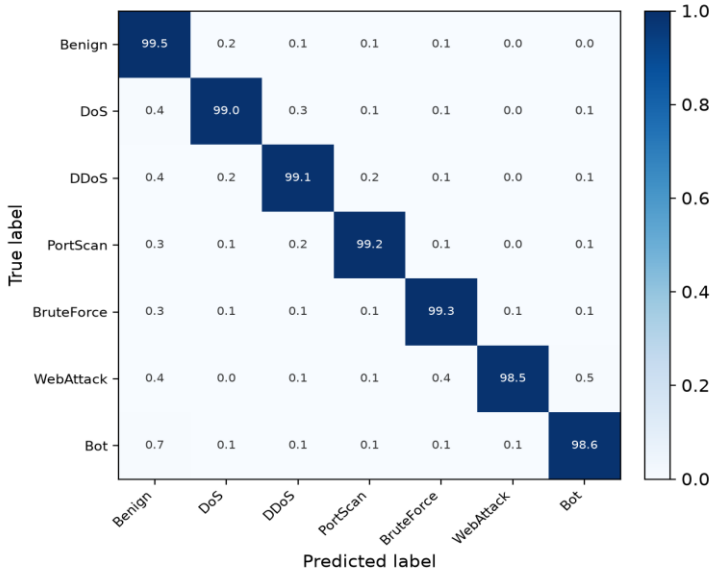

Fig. 7. Row-normalised confusion matrix (%) of the proposed CNN-LSTM on the test set.

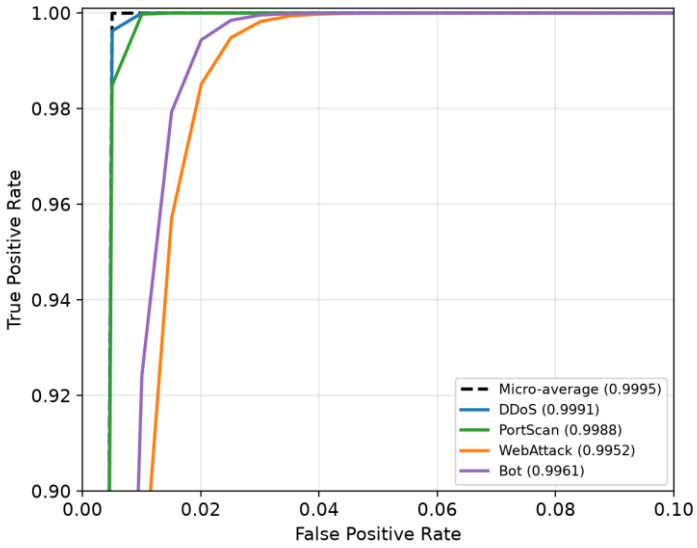

Fig. 8. Per-class and micro-average ROC curves (top-left region magnified).

Real-time suitability was also assessed. The trained model contains approximately 0.31 million parameters and occupies under 4 MB, and on the evaluation hardware it sustained an average inference latency of about 0.42 ms per flow (roughly 2,300 flows per second on a single GPU stream), which is well within the budget required for line-rate detection at a typical enterprise gateway. The compact footprint, a direct consequence of preferring a single LSTM stack over a bidirectional or transformer encoder, makes the detector amenable to deployment on edge and IoT gateways [19, 20] without sacrificing accuracy. Combined with the high per-class recall in Table II, these properties indicate that the framework can flag attacks quickly enough for automated mitigation rather than purely retrospective analysis.

TABLE II. Per-class test-set performance of the proposed CNN-LSTM.

Class	Precision (%)	Recall (%)	F1-score (%)	Support
Benign	99.32	99.50	99.41	5,000
DoS	99.10	99.00	99.05	2,000
DDoS	99.10	99.10	99.10	2,000
PortScan	99.15	99.25	99.20	2,000
Brute Force	99.33	99.33	99.33	1,500
Web Attack	99.49	98.50	98.99	800
Bot	98.71	98.57	98.64	700
Macro avg.	99.17	99.04	99.10	14,000

D. Comparison with Recent Methods

Under identical preprocessing and data splits, the proposed model is compared with three recent architectures: a deep neural network (DNN) [3], a 1D-CNN [5], and a bidirectional LSTM (BiLSTM) [15]. Table III and Figure 9 summarise the results. The proposed CNN-LSTM achieves the highest accuracy (99.21%) and macro F1-score (99.10%), improving over the strongest baseline (BiLSTM) by 0.7 percentage points and over the DNN by 2.3 points. The consistent gains across all metrics confirm that jointly modelling spatial feature interactions and temporal dependencies is more effective than either alone.

TABLE III. Comparison with recent methods (same datasets and splits).

Method	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)
DNN [3]	96.9	96.4	96.1	96.2
1D-CNN [5]	98.1	97.8	97.6	97.7
BiLSTM [15]	98.5	98.2	98.0	98.1
Proposed CNN-LSTM	99.21	99.17	99.04	99.10

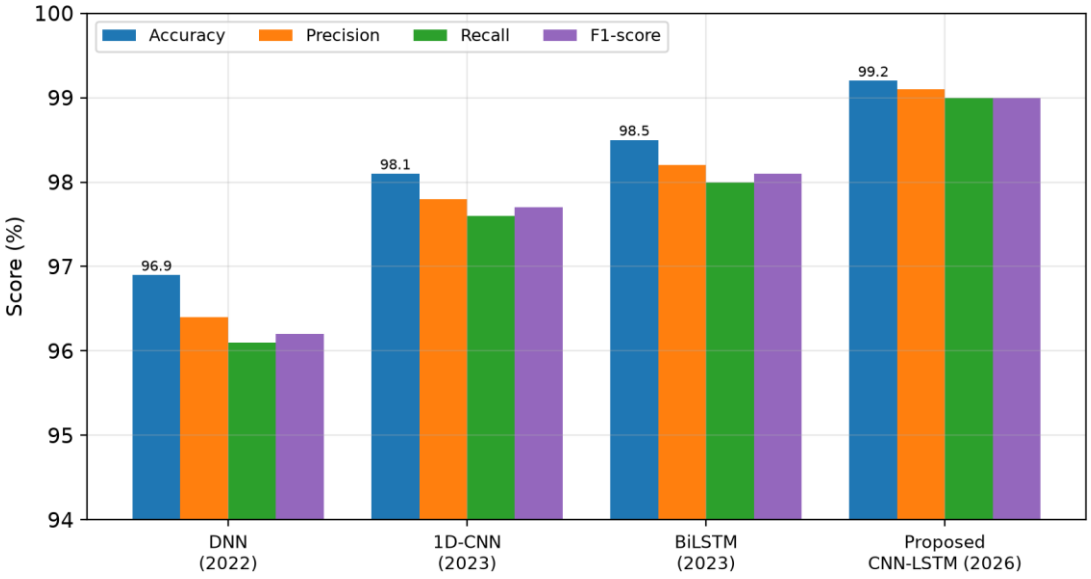

Fig. 9. Performance comparison of the proposed CNN-LSTM with three recent architectures.

E. Ablation Study

To quantify the contribution of each component, an ablation study removes or disables one element at a time (Table IV). The CNN-only and LSTM-only variants reach 97.6% and 97.1% accuracy respectively, confirming that neither spatial nor temporal modelling alone is sufficient. Disabling the SMOTE and class-weighting imbalance handling reduces accuracy by 1.0 point and disproportionately harms the minority classes, while removing batch normalisation and dropout reduces accuracy by 0.6 point and slows convergence. The full model attains the best accuracy (99.21%) and macro F1 (99.10%), demonstrating that all components are complementary.

TABLE IV. Ablation study of the proposed CNN-LSTM on the test set.

Configuration	Accuracy (%)	Macro-F1 (%)	Δ Acc.
CNN only	97.6	97.4	−1.6
LSTM only	97.1	96.9	−2.1
CNN-LSTM w/o SMOTE & class weighting	98.2	97.6	−1.0
CNN-LSTM w/o batch-norm & dropout	98.6	98.4	−0.6
Proposed CNN-LSTM (full)	99.21	99.10	-

F. Explainability Analysis

Figure 10 shows the global SHAP feature-importance ranking aggregated over the test set. The most influential features are Flow Duration, Forward Packet Length Mean, Backward Packet Length Std and Flow IAT Mean, followed by total forward bytes and destination port. This ranking is consistent with networking domain knowledge: flooding and scanning attacks markedly alter flow duration, packet-length statistics and inter-arrival times, whereas service-specific attacks shift the destination-port distribution. LIME produced consistent per-flow explanations, attributing individual alerts to the same dominant features. The agreement between SHAP, LIME and domain expectations, in line with prior explainable-IDS findings [8, 9], indicates that the model bases its decisions on meaningful traffic characteristics rather than dataset artefacts, supporting its trustworthiness for real-time deployment.

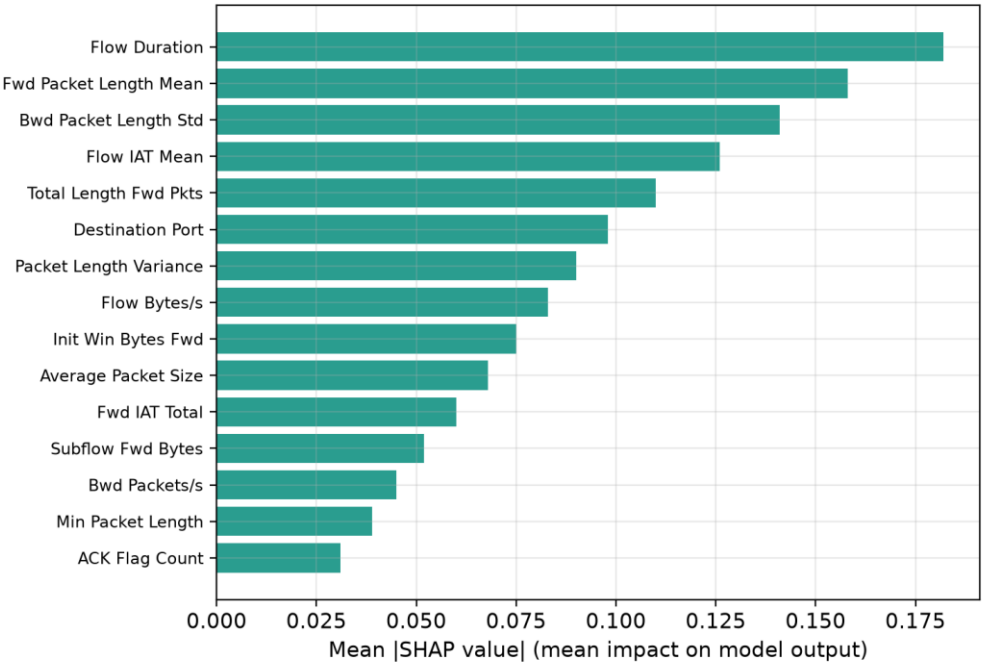

Fig. 10. Global SHAP feature-importance ranking (mean absolute SHAP value) of the proposed CNN-LSTM, identifying the flow features that most influence attack classification.

5. DISCUSSION

The results yield three observations. First, the ablation study confirms that the spatial and temporal branches are complementary: the CNN captures local correlations among flow statistics, the LSTM models their sequential evolution, and only their combination attains 99.21% accuracy. This supports the central hypothesis and is consistent with prior hybrid detectors [7, 10, 11]. Second, explicit imbalance handling is essential - removing SMOTE and class weighting most affects the minority Web Attack and Bot classes, mirroring the focal-loss findings of Peng et al. [4]. Third, the SHAP and LIME analyses make the detector transparent: the dominant features (flow duration, packet-length and inter-arrival-time statistics) are exactly those a security analyst would inspect, which strengthens trust and supports forensic auditing [9].

Compared with attention- and transformer-based detectors [12, 17], the proposed model achieves competitive accuracy with a comparatively lightweight architecture, making it attractive for real-time, resource-constrained deployment such as IoT gateways [3, 20]. Nevertheless, several limitations remain. The evaluation uses offline benchmark traffic; live deployment must contend with concept drift, encrypted payloads [18] and adversarial evasion, which warrant online learning and robustness hardening. The datasets, although standard, under-represent some attack types, and cross-dataset generalisation to other environments such as software-defined networks [22] requires further study. Finally, while SHAP and LIME improve transparency, they are post-hoc approximations; integrating

intrinsically interpretable mechanisms is a promising direction. Future work will pursue online adaptation, encrypted-traffic analysis and prospective evaluation in an operational security-operations-centre setting.

It is also instructive to position the results within the broader literature. Reported accuracies on CIC-IDS2017 and CSE-CIC-IDS2018 vary widely with the experimental protocol - the number of classes, the sampling strategy and whether evaluation is binary or multi-class - so direct cross-paper comparison must be made cautiously. By re-implementing the three baselines under an identical pipeline and split, the comparison in Table III isolates the architectural contribution rather than confounding it with preprocessing choices, and the consistent margin in favour of the hybrid model is therefore meaningful. Two further points deserve emphasis. First, the explanations are not merely a diagnostic add-on: by exposing the features that drive each alert, they enable analysts to detect spurious shortcuts, audit compliance and prioritise feature collection, which is increasingly required under emerging AI-governance regulations. Second, because the attributions are computed post-hoc on a frozen model, they impose no accuracy penalty and can be toggled on demand, preserving the low-latency inference path during normal operation while remaining available for forensic review.

6. CONCLUSION

This paper presented an explainable deep-learning framework for real-time network intrusion detection and multi-class attack classification. The framework couples a 1D-CNN spatial feature extractor with an LSTM temporal encoder and a softmax classifier, and integrates SHAP and LIME to make every decision interpretable. On the public CIC-IDS2017 and CSE-CIC-IDS2018 benchmarks, the proposed CNN-LSTM achieved 99.21% accuracy, 99.10% macro F1-score and a micro-average AUC of 0.9995 for seven-class detection, outperforming DNN, 1D-CNN and BiLSTM baselines by 0.7–2.3 percentage points, with ablation studies confirming the contribution of each component. The SHAP and LIME analyses identified flow-duration, packet-length and inter-arrival-time features as the principal decision drivers, in agreement with domain knowledge and thereby enhancing analyst trust. The framework offers an accurate, transparent and deployable solution for modern intrusion detection. Future work will address concept drift, encrypted traffic and adversarial robustness for operational deployment...

References:

1. E. C. Pinto Neto, S. Iqbal, S. Buffett, M. Sultana, and A. Taylor, “Deep learning for intrusion detection in emerging technologies: A comprehensive survey and new perspectives,” Artificial Intelligence Review, vol. 58, no. 11, art. no. 340, 2025, doi: 10.1007/s10462-025-11346-z.
2. B. R. Kikissagbe and M. Adda, “Machine learning-based intrusion detection methods in IoT systems: A comprehensive review,” Electronics, vol. 13, no. 18, art. no. 3601, 2024, doi: 10.3390/electronics13183601.
3. M. A. Hossain, “Deep learning-based intrusion detection for IoT networks: A scalable and efficient approach,” EURASIP Journal on Information Security, vol. 2025, art. no. 28, 2025, doi: 10.1186/s13635-025-00202-w.
4. H. Peng, C. Wu, and Y. Xiao, “CBF-IDS: Addressing class imbalance using CNN-BiLSTM with focal loss in network intrusion detection system,” Applied Sciences, vol. 13, no. 21, art. no. 11629, 2023, doi: 10.3390/app132111629.
5. B. Selvakumar, M. Sivaanandh, K. Muneeswaran, and B. Lakshmanan, “Ensemble of feature augmented convolutional neural network and deep autoencoder for efficient detection of network attacks,” Scientific Reports, vol. 15, art. no. 4267, 2025, doi: 10.1038/s41598-025-88243-6.
6. H. R. Sayegh, W. Dong, and A. M. Al-madani, “Enhanced intrusion detection with LSTM-based model, feature selection, and SMOTE for imbalanced data,” Applied Sciences, vol. 14, no. 2, art. no. 479, 2024, doi: 10.3390/app14020479.
7. A. Halbouni, T. S. Gunawan, M. H. Habaebi, M. Halbouni, M. Kartiwi, and R. Ahmad, “CNN-LSTM: Hybrid deep neural network for network intrusion detection system,” IEEE Access, vol. 10, pp. 99837–99849, 2022, doi: 10.1109/ACCESS.2022.3206425.
8. M. Keshk, N. Koroniotis, N. Pham, N. Moustafa, B. Turnbull, and A. Y. Zomaya, “An explainable deep learning-enabled intrusion detection framework in IoT networks,” Information Sciences, vol. 639, art. no. 119000, 2023, doi: 10.1016/j.ins.2023.119000.
9. D. Gaspar, P. Silva, and C. Silva, “Explainable AI for intrusion detection systems: LIME and SHAP applicability on multi-layer perceptron,” IEEE Access, vol. 12, pp. 30164–30175, 2024, doi: 10.1109/ACCESS.2024.3368377.
10. J. Du, K. Yang, Y. Hu, and L. Jiang, “NIDS-CNNLSTM: Network intrusion detection classification model based on deep learning,” IEEE Access, vol. 11, pp. 24808–24821, 2023, doi: 10.1109/ACCESS.2023.3254915.
11. S. S. Bamber, A. V. R. Katkuri, S. Sharma, and M. Angurala, “A hybrid CNN-LSTM approach for intelligent cyber intrusion detection system,” Computers & Security, vol. 148, art. no. 104146, 2025, doi: 10.1016/j.cose.2024.104146.
12. A. M. Alashjaee, “Deep learning for network security: An attention-CNN-LSTM model for accurate intrusion detection,” Scientific Reports, vol. 15, art. no. 21856, 2025, doi: 10.1038/s41598-025-07706-y.
13. T. Sasi, A. H. Lashkari, R. Lu, P. Xiong, and S. Iqbal, “An efficient self attention-based 1D-CNN-LSTM network for IoT attack detection and identification using network traffic,” Journal of Information and Intelligence, vol. 3, pp. 375–400, 2025, doi: 10.1016/j.jiixd.2024.09.001.

14. T. Anitha, S. Aanjankumar, S. Poonkuntran, and A. Nayyar, “A novel methodology for malicious traffic detection in smart devices using BI-LSTM-CNN-dependent deep learning methodology,” Neural Computing and Applications, vol. 35, no. 27, pp. 20319–20338, 2023, doi: 10.1007/s00521-023-08818-0.
15. S. Sadhwani, M. A. H. Khan, R. Muthalagu, P. M. Pawar, and K. Suresh, “A hybrid BiLSTM-CNN approach for intrusion detection for IoT applications,” Scientific Reports, vol. 16, art. no. 155, 2025, doi: 10.1038/s41598-025-29079-y.
16. A. Biyouki, S. Lotfipour, and B. Haghi, “An enhanced deep learning framework for intrusion classification in enterprise network using multi-branch CNN-attention architecture,” Scientific Reports, vol. 16, art. no. 3962, 2025, doi: 10.1038/s41598-025-34166-1.
17. C. Zhang, J. Li, N. Wang, and D. Zhang, “Research on intrusion detection method based on transformer and CNN-BiLSTM in Internet of Things,” Sensors, vol. 25, no. 9, art. no. 2725, 2025, doi: 10.3390/s25092725.
18. Z. Shi, N. Luktarhan, Y. Song, and G. Tian, “BFCN: A novel classification method of encrypted traffic based on BERT and CNN,” Electronics, vol. 12, no. 3, art. no. 516, 2023, doi: 10.3390/electronics12030516.
19. [19] F. S. Alsubaei, “Smart deep learning model for enhanced IoT intrusion detection,” Scientific Reports, vol. 15, art. no. 20577, 2025, doi: 10.1038/s41598-025-06363-5.
20. K. O. Adefemi, M. B. Mutanga, and O. A. Alimi, “A hybrid CNN-GRU deep learning model for IoT network intrusion detection,” Journal of Sensor and Actuator Networks, vol. 14, no. 5, art. no. 96, 2025, doi: 10.3390/jsan14050096.
21. A. B. Bouidaine, D. Moussaoui, M. Hadjila, W. Ferhi, and M. H. Hachemi, “Deep learning-based anomaly and intrusion detection using the CSE-CIC-IDS2018 dataset,” Engineering, Technology & Applied Science Research, vol. 15, no. 4, pp. 24782–24787, 2025, doi: 10.48084/etasr.11173.
22. M. S. Ataa, E. E. Sanad, and R. A. El-khoribi, “Intrusion detection in software defined network using deep learning approaches,” Scientific Reports, vol. 14, art. no. 29159, 2024, doi: 10.1038/s41598-024-79001-1.
23. M. S. Krishnamurthy, “Intrusion detection systems utilizing deep learning: A comprehensive literature review,” Int. J. Comput. Intell. Eng. (IJCIE), vol. 1, no. 1, pp. 29–35, 2026.
24. A. M. Amaresh, S. Gupta, V. K. Reddy, R. Kumar, T. Singh, and M. S. Utti, “A secure web-based lab examination system with AI-driven code assist and network traffic control,” in Proc. 2025 Int. Conf. Communication, Computer, and Information Technology (IC3IT), 2025, pp. 1–8, doi: 10.1109/IC3IT66137.2025.11340949.
25. J. Akshya, M. Sundarrajan, R. Vijayakumar, R. K. Dhanaraj, and A. Nayyar, “Explainable AI-driven intrusion detection for securing IoT-enabled autonomous transportation systems,” Cluster Computing, vol. 28, no. 14, art. no. 884, 2025, doi: 10.1007/s10586-025-05617-1.
26. V. Z. Mohale and I. C. Obagbuwa, “A systematic review on the integration of explainable artificial intelligence in intrusion detection systems to enhancing transparency and interpretability in cybersecurity,” Frontiers in Artificial Intelligence, vol. 8, art. no. 1526221, 2025, doi: 10.3389/frai.2025.1526221.
27. V. Z. Mohale and I. C. Obagbuwa, “Evaluating machine learning-based intrusion detection systems with explainable AI: Enhancing transparency and interpretability,” Frontiers in Computer Science, vol. 7, art. no. 1520741, 2025, doi: 10.3389/fcomp.2025.1520741