Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

# A CNN–LSTM Hybrid Model for Reducing False Positives in Anomaly-Based Intrusion Detection Systems

**Sameer Tembhurney**
Bhabha University
**Tripti Arjariya**
info@bhabhauniversity.edu.in

Bhabha University

**Research Article**

**Keywords:** Intrusion Detection, Anomaly Detection, CNN, LSTM, False Positives, Net- work Security

**Posted Date:** May 8th, 2026

**DOI:** https://doi.org/10.21203/rs.3.rs-9244380/v1

**License:** This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

**Additional Declarations:** No competing interests reported.

# A CNN–LSTM Hybrid Model for Reducing False Positives in Anomaly-Based Intrusion Detection Systems

**Sameer Tembhurney**
Department of Computer Science and Engineering
Bhabha University, Bhopal, Madhya Pradesh 462026, India
Email: sameer.tembhurney@gmail.com

**Tripti Arjariya***
Principal and Professor, Department of Computer Science and Engineering
Bhabha University, Bhopal, Madhya Pradesh 462026, India
Email: info@bhabhauniversity.edu.in

*Corresponding Author

**Abstract**

The rapid expansion of next-generation networks has heightened the risk of sophisticated cyber intrusions, making anomaly-based intrusion detection systems (IDS) a critical component of modern cybersecurity. Traditional IDS techniques, while effective in detecting known attacks, often suffer from high false positive rates and limited adaptability to evolving threats. To address these challenges, this paper proposes a hybrid deep learning framework that integrates Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) models for efficient intrusion detection. CNN layers are employed to extract spatial patterns from traffic features, while LSTM units capture temporal dependencies in sequential data, enabling the system to detect both short-lived and long-range anomalies. To further enhance generalization, regularization strategies and post-processing mechanisms are incorporated for false positive suppression. The proposed model is evaluated on benchmark datasets such as NSL-KDD, CIC-IDS2017, and UNSW-NB15, achieving significant improvements in accuracy, detection rate, and false positive reduction compared to conventional machine learning and deep learning baselines. The results demonstrate that the CNN–LSTM hybrid model provides a scalable and intelligent IDS solution, capable of strengthening network defense mechanisms for next-generation autonomous and heterogeneous environments.

**Keywords:** Intrusion Detection, Anomaly Detection, CNN, LSTM, False Positives, Network Security

# 1 Introduction

With the evolution of next-gen communication networks, the Internet of Things, and massive cloud service landscapes, the current digital landscape has drastically changed [1–3]. . Billions of devices across the globe are currently connected and communicating in real-time, which has made cybersecurity the next big challenge, as it is essential for the security of the network and its resilience [3–7]. According to industry analyses, not only is the amount of cyber threats rising exponentially, but the complexity of these threats is also building up. Advanced persistent threats, ransomware, and zero-day attacks are currently affecting the commercial and industrial sectors [8–11]. Moreover, the traditional perimeter security solutions, including firewalls and signature-based intrusion detection solutions, are now widely failing as these solutions are based on known patterns [12–16]. Anomaly-based intrusion detection has hence emerged as a necessity for identifying new and unknown techniques of attack by identifying abnormalities in network behavior [8, 15, 17–19].

Despite the strengths of anomaly-based intrusion detection systems (IDS), they continue to struggle with two significant challenges: too many false alarms, which overwhelm security personnel, and lackluster accuracy in detecting minority classes of attacks [8, 13, 14, 19, 20]. The former is a problem because too many false alarms trigger alert fatigue, while the latter is a nuisance because it puts many systems at risk, compromising their security, confidentiality, and availability [3, 10, 11, 13]. To solve these problems, anomaly-based intrusion detection systems need to be smart, not only raising the accuracy rate but also significantly lowering the rates of false alarms [4, 11, 12, 19, 21]. Lately, artificial intelligence and deep learning techniques have been found to be quite useful in addressing these problems [11, 22–25].

Deep learning architectures, for example, such as Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN), have achieved spectacular results in various fields, including computer vision, natural language processing, and time series analysis, etc [1, 22, 26]. In regard to Intrusion Detection Systems, CNN is very good at extracting spatial features from network data, and Long Short-Term Memory Networks (LSTM), which is a subset of RNN, can interpret the evolution of data over time for sequential data of network flows [25,27–31]. However, most of the related studies focused only on using CNN or LSTM for an Intrusion Detection System, without exploring completely the potential of combined spatial and temporal features [23, 27, 32, 33]. In addition, there exist many challenges for using deep learning, e.g., class imbalance, scalability, and generalization over different environments, etc [13, 19–21, 34].

To address the existing gap in traditional intrusion detection mechanisms, a novel hybrid structure of a deep learning model is presented that leverages the combined potency of both CNN and LSTM architectures in anomaly-based intrusion detection [32, 35, 36]. Specifically, the CNN component is utilized to identify both low-level and high-level spatial features from the traffic flows within a computer network [27, 32, 37]. Meanwhile, the LSTM component is leveraged to identify temporal features and dependencies, thereby allowing long-range relationships between anomalies to be determined [23, 25, 35, 36, 38]. To detect any false positives in the output of the detection system, regularization procedures, thresholding, smoothing operations, and even the suppression of false alarms are utilized [4,11,21]. Overall, all the aspects of the system are appropriately balanced so that no critical network threats are overlooked [11, 13, 15].

The model is thoroughly tested with three well-known benchmark datasets, namely, NSL-KDD, CIC-IDS2017, and UNSW-NB15 [8, 39–41]. The datasets provide a range of traffic patterns while covering a significant range of intrusion types, including Denial of Service (DoS), User to Root (U2R), Remote to Local (R2L), as well as probing attacks [42–46]. From the experiments, it is clear that the CNN-LSTM hybrid architecture outperforms yet again the traditional machine learning approaches like SVM, Random Forest, XGBoost, as well as contemporary deep learning approaches like CNN or LSTM [23, 38]. In particular, it outperforms these approaches in terms of accuracy, precision, recall, as well as F1-score, while at the same time significantly reducing the rate of false alarm, thereby validating the potentials of this intrusion detection framework in enhancing the accuracy of intrusion detection systems for next-generation networks [27, 33, 35] [3, 4, 19, 21]

The main contributions of this work can be summarized in the following points:

- Our paper proposes a novel CNN-LSTM hybrid framework that incorporates the benefits of spatial and temporal feature learning for the purpose of anomaly-based intrusion detection.
- The problem of false positives is addressed by the use of regularization, adaptive thresholding, and post-processing techniques.
- We design comprehensive experiments using common benchmark data sets like NSL-KDD, CIC-IDS2017, UNSW-NB15 to assess the effectiveness of the model as well as its generality.
- The results demonstrate a significant improvement in detection accuracy and operational efficiency by the hybrid technique and highlight its prospects for future applications in next-generation autonomous and diverse networks.

# 2 Related Work

Intrusion Detection Systems have been in the forefront for decades, especially as the need for increased security from cyber attacks becomes more critical in the lives of organizations, countries, and individuals in general [14–16, 18]. Broadly speaking, there exist two types of intrusion detection systems, namely signature-based intrusion detection systems and anomaly-based intrusion detection systems. The former comprise the system's ability to monitor the entering traffic pattern in the system against a list of known attacks or fingerprints of the attackers [14]. The signature-based system can detect attacks that have already been documented, thus acting as an effective barrier against such attacks. However, it might not be effective in detecting unknown attacks or the so-called zero-day attacks. The anomaly-based intrusion detection system, on the other hand, attempts to identify the kind of acceptable behavior within the system, detecting any abnormalities [18, 19]. The drawback to anomaly-based intrusion detection systems is the high number of false alarms or positives, an issue that has prompted further research in interpreting the results, as it affects the accuracy of the system.

Anomaly-based detection systems have historically been an important part of the arsenal of traditional detection systems, relying upon statistical modeling procedures, rule-based

systems, and clustering techniques to try to develop an impression of what constituted 'normal' network operation [18]. Unfortunately, such techniques have only really been able to hold up well in more structured and simplified environments; as networks became grown disorderly, complicated, and speeded-up, more standardized statistical measurements were less reliable than ever before.

To extend the existing limits, the field of machine learning has been employed. The tools used from this arsenal include the likes of Support Vector Machines, Decision Trees, Random Forests, and k-Nearest Neighbors [19]. They were deemed effective in discriminating normal from anomalous data because they learned from labeled examples. However, their usage has been hindered by the need for labeled features, and effectiveness decreases in the case of data imbalance [20].

The problem of false positives stubbornly persists and lies at the very heart of intrusion detection system research. When alerts come flooding in, confidence in the system is rapidly lost and admins are left wading through time-consuming and expensive verifications. To blunt this problem, various approaches have been tested, including ensemble methods, careful feature selection, adaptive thresholding, and cost-sensitive classification [19].. While these ideas have brought minor gains, they often come at the expense of sensitivity or by adding computational loads. Many models also perform terribly when networks alter conditions or attackers change tactics. That's why more strong and generalizable architectures for intrusion detection systems have taken on a sense of urgency.

Then deep learning introduced a completely new direction to IDS, allowing models to learn features directly from traffic data, even if raw or only lightly preprocessed [23, 24]. Convolutional Neural Networks have been very successful in extracting spatial patterns from network traffic with improved accuracy and robustness compared to traditional machine learning. However, sometimes CNNs lose the temporal relationships that are encoded within flow data. In response, researchers adopted recurrent architectures such as Long Short-Term Memory networks, and these appear particularly effective at capturing temporal dependencies across sequences of traffic. LSTMs are particularly effective in detecting attacks with long-term dependences; consider stealthy scans or wide-ranging DDoS attacks. As a standalone solution, however, LSTMs may be computationally expensive and can easily incur overfitting issues when noise or class imbalance is severe in the data [20].

To integrate the best of all worlds, CNN–LSTM hybrids have surfaced, with CNNs dealing with spatial patterns and LSTMs dealing with temporal sequence data, thereby providing end-to-end coverage of network traffic patterns [31, 32, 38]. This has brought hybrid models from literature, like Li et al., Hariharan et al., and Wu et al., which give an overview of local patterns and their interdependence, thereby improving detection efficiency. CNN–LSTM hybrid models have, for benchmark purposes, managed to raise the bar compared to other deep learning models in terms of accuracy and detection rate for different types of network attacks [31, 32, 38]. However, when applied in real-world scenarios, they still pose to be scalability and false alarm threats.

The body of literature on IDS shows that while significant achievements have been realized, there are certain challenges to be addressed [15]. For instance, there have always been challenges with the number of false positives in anomaly-based IDS systems. In terms of machine learning and deep learning approaches, they have been robust but have difficulties with class imbalance problems, scalability, as well as coping with real-world scenarios [19]. While

it is promising with hybrid models like CNN–LSTM architectures, it still needs to be perfected before it can find actual application in next-generation networks as an anomaly-based IDS system. For instance, there have always been challenges with false positives; hence, there is a need to address this gap with certain false-positive reduction strategies like temporal smoothing that could find application in next-generation networks as an anomaly-based IDS system. This gap forms the core motivation for this study because we propose to create a model that has false-positive reduction strategies using a hybrid model like CNN–LSTM.

# 3 Preliminaries

In this section, we introduce the mathematical formulation of the anomaly-based intrusion detection problem, define the notations used throughout the paper, and present the evaluation metrics that guide our analysis.

## Notation

Let $X \in \mathbb{R}^{n \times d}$ denote the network traffic dataset, where $n$ represents the number of samples and $d$ is the dimensionality of features. Each instance is represented as a feature vector:

$$x_i = (x_{i1}, x_{i2}, \ldots, x_{id}), \quad x_i \in \mathbb{R}^d, \tag{1}$$

where $x_{ij}$ corresponds to the $j$-th feature of the $i$-th sample. The corresponding ground truth label is denoted by:

$$y_i \in \{0, 1\}, \tag{2}$$

where $y_i = 0$ represents benign traffic and $y_i = 1$ represents malicious traffic.

For sequential modeling, we represent the temporal dimension by grouping network flows into windows of size $T$. Thus, a sequence of traffic events can be written as:

$$\mathbf{X}_i = \{x_{i1}, x_{i2}, \ldots, x_{iT}\}, \quad \mathbf{X}_i \in \mathbb{R}^{T \times d}. \tag{3}$$

## Problem Formulation

The anomaly detection task can be defined as learning a function:

$$f : \mathbb{R}^{T \times d} \to \{0, 1\}, \tag{4}$$

that maps a sequence of traffic features $\mathbf{X}_i$ to a binary decision.

The CNN component extracts spatial features:

$$z_i^{\text{CNN}} = f_{\text{CNN}}(\mathbf{X}_i), \tag{5}$$

while the LSTM component captures temporal dependencies:

$$z_i^{\text{LSTM}} = f_{\text{LSTM}}(z_i^{\text{CNN}}). \tag{6}$$

The fused representation is then passed through a classifier to produce the predicted probability of anomaly:

$$\hat{y}_i = g(z_i^{\text{LSTM}}), \qquad \hat{y}_i \in [0, 1]. \tag{7}$$

The final decision rule is obtained by applying a threshold $\tau \in [0, 1]$:

$$\tilde{y}_i = \begin{cases} 1, & \text{if } \hat{y}_i \geq \tau, \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

## Evaluation Metrics

To evaluate detection performance, we define the confusion matrix elements: TP (True Positives), TN (True Negatives), FP (False Positives), FN (False Negatives).

The standard metrics are defined as:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}, \tag{9}$$

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall (TPR)} = \frac{TP}{TP + FN}, \tag{10}$$

$$\text{F1-score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{11}$$

$$\text{False Positive Rate (FPR)} = \frac{FP}{FP + TN}. \tag{12}$$

For robust comparison, ranking-based metrics are also reported:

$$\text{ROC–AUC} = \int_0^1 TPR(FPR)\, dFPR, \tag{13}$$

$$\text{PR–AUC} = \int_0^1 \text{Precision}(\text{Recall})\, d\text{Recall}. \tag{14}$$

## Class Imbalance and Cost Sensitivity

In real-world IDS datasets, attack samples are often much fewer than normal traffic samples, creating a strong class imbalance. To handle this, cost-sensitive learning introduces class weights $\alpha_0$ and $\alpha_1$ into the loss function:

$$L = -\sum_{i=1}^{5} \alpha_1 y_i \log(\hat{y}_i) + \alpha_0 (1 - y_i) \log(1 - \hat{y}_i)^{6}, \tag{15}$$

where $\alpha_1 > \alpha_0$ penalizes misclassification of attacks more heavily, thereby reducing false negatives while maintaining manageable false positives.

# 4 Proposed Method

The current section briefly presents the architecture, design rationale, as well as the entire workflow of the proposed model, i.e., CNN-LSTM Hybrid Architecture. The model leverages both spatial and temporal perspectives of network communication to improve its accuracy as well as minimize false alarms. Figure 1 provides an overall idea about the proposed ANTS system, which covers data preprocessing, feature extraction by applying CNN, sequence modeling by applying LSTM, and finally, postprocessing to minimize false alarms.

## 4.1 System Overview

The overall system architecture consists of four main components:

1. **Data preprocessing and feature engineering**, which converts raw network traffic into structured tensors suitable for deep learning models.
2. **CNN backbone**, which extracts spatial correlations and localized feature hierarchies from the input features.
3. **LSTM temporal modeling module**, which captures sequential dependencies and long-range attack patterns over time.
4. **Post-processing and calibration**, which refines prediction outputs using temporal smoothing and adaptive thresholding to minimize false alarms.

Each network flow is represented as a sequence of $T$ feature vectors of dimension $d$, forming a 2D matrix $\mathbf{X} \in \mathrm{R}^{T \times d}$. The CNN module processes $\mathbf{X}$ to learn spatial dependencies among correlated features, producing intermediate feature maps. These representations are then passed to the LSTM module to model temporal relations among consecutive traffic windows, yielding a discriminative sequence embedding $\mathbf{h}_T$ . The final classifier predicts the probability of an intrusion using a sigmoid activation layer.

## 4.2 Data Pipeline and Feature Engineering

### 4.2.1 Traffic Parsing and Normalization

Network traffic records from datasets such as NSL-KDD, CIC-IDS2017, and UNSW-NB15 are parsed into feature matrices. Continuous numerical features are normalized using min–max scaling:

$$x_{ij}^{\text{norm}} = \frac{x_{ij} - \min(x_j)}{\max(x_j) - \min(x_j)'} \tag{16}$$

ensuring stable convergence during gradient optimization. Features with large variance (e.g., byte counts or duration) are log-transformed to mitigate skewness.

### 4.2.2 Temporal Windowing and Sequence Construction

Packets or flows are grouped into fixed-length time windows of size $T$ based on arrival order or timestamps. Each sequence $\mathbf{X}_i = [x_{i1}, x_{i2}, \ldots, x_{iT}]$ captures short-term and long-term traffic dependencies, essential for detecting multi-stage attacks such as DDoS or probe scans.

### 4.2.3 Categorical Encoding and Scaling

Categorical fields (e.g., protocol type, service, flag) are encoded using one-hot or label encoding, depending on dimensionality. Feature standardization ensures zero mean and unit variance before feeding the model.

## 4.3 CNN Backbone for Spatial Feature Extraction

The CNN module acts as a local pattern detector, identifying feature correlations such as port–protocol–flag combinations that characterize attack signatures. The input $\mathbf{X}_i$ is reshaped into a pseudo-image $\mathbf{I}_i \in \mathbb{R}^{T \times d \times 1}$, and multiple 1D or 2D convolutional layers are applied:

$$z_k = \sigma(\mathbf{W}_k * \mathbf{I}_i + b_k), \tag{17}$$

where $\mathbf{W}_k$ and $b_k$ denote the convolution kernel and bias, and $\sigma(\cdot)$ is the ReLU activation function. Pooling layers (max or average pooling) reduce dimensionality and emphasize salient features. Batch normalization is used to stabilize activations and accelerate training.

Residual connections may optionally be introduced to improve gradient flow:

$$z^{(l+1)} = z^{(l)} + F(z^{(l)}, \mathbf{W}^{(l)}), \tag{18}$$

where $F(\cdot)$ represents the convolution–activation sequence in layer $l$.

## 4.4 LSTM Module for Temporal Modeling

The CNN-derived feature sequence is flattened into a temporal representation and passed to the LSTM module, which models inter-frame correlations across time steps:

$$i_t = \sigma(W_i z_t + U_i h_{t-1} + b_i), \tag{19}$$
$$f_t = \sigma(W_f z_t + U_f h_{t-1} + b_f), \tag{20}$$
$$o_t = \sigma(W_o z_t + U_o h_{t-1} + b_o), \tag{21}$$
$$\tilde{c}_t = \tanh(W_c z_t + U_c h_{t-1} + b_c), \tag{22}$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t, \tag{23}$$
$$h_t = o_t \odot \tanh(c_t), \tag{24}$$

where $i_t$, $f_t$, and $o_t$ denote the input, forget, and output gates, respectively. The final hidden state $h_T$ encodes both short- and long-term dependencies. For richer context, a bidirectional LSTM (BiLSTM) may be used, concatenating forward and backward hidden states:

$$h_T = [\overrightarrow{h_T}; \overleftarrow{h_T}]. \tag{25}$$

## 4.5 Feature Fusion and Classifier Head

CNN and LSTM features can be fused in two configurations:

- **Serial fusion (CNN → LSTM)**: CNN features are sequentially fed into the LSTM network, allowing temporal reasoning over spatially extracted patterns.
- **Parallel fusion**: CNN and LSTM operate independently on the same input, and their outputs are concatenated before classification:

$$z_{\text{fusion}} = [z^{\text{CNN}}; z^{\text{LSTM}}], \tag{26}$$

  followed by dense layers and dropout regularization.

The final classifier applies a fully connected layer with sigmoid activation:

$$\hat{y} = \sigma(\mathbf{w}^{\top} z_{\text{fusion}} + b). \tag{27}$$

## 4.6 Loss Functions and Optimization

Given class imbalance and asymmetric misclassification costs, we adopt a weighted cross-entropy loss:

$$L_{\text{WCE}} = -\sum_{i} [\alpha_1 y_i \log(\hat{y}_i) + \alpha_0 (1 - y_i) \log(1 - \hat{y}_i)], \tag{28}$$

where $\alpha_1 > \alpha_0$ penalizes missed attacks more heavily. Optionally, focal loss is used to emphasize hard examples:

$$L_{\text{Focal}} = -\sum_{i} (1 - \hat{y}_i)^{\gamma} y_i \log(\hat{y}_i). \tag{29}$$

Model parameters are optimized using the Adam optimizer with learning rate scheduling and early stopping based on validation loss.

## 4.7 Regularization and Generalization

To prevent overfitting and enhance generalization:

- Dropout layers (rate = 0.3–0.5) are applied between dense layers.
- L2 weight decay is used on all convolutional and recurrent kernels.
- Early stopping halts training when the validation loss plateaus for $k$ epochs.

Advanced data-augmentation techniques such as Mixup and CutMix can be adapted to tabular/temporal inputs by linear interpolation of consecutive flows.

## 4.8 Calibration and Thresholding

The raw sigmoid outputs are calibrated using Platt scaling or isotonic regression to improve probability reliability. The optimal classification threshold $\tau^*$ is chosen via Youden's J statistic:

$$\tau^* = \arg\max_{\tau}(TPR(\tau) - FPR(\tau)), \tag{30}$$

subject to operational constraints (e.g., FPR ≤ 0.02). This ensures balanced sensitivity and specificity.

## 4.9 Post-Processing for False-Positive Reduction

To mitigate alarm flooding, we apply:

1. **Temporal smoothing:** majority voting across a sliding window of recent predictions.
2. **Alarm suppression windows:** suppress duplicate alarms within a fixed time horizon $\Delta t$ for identical source–destination pairs.

This step reduces redundant alerts from transient anomalies without compromising detection of sustained attacks.

## 4.10 Complexity Analysis

Let $n$ denote the number of samples, $T$ the sequence length, $d$ the feature dimension, $k$ the kernel size, and $h$ the LSTM hidden units. The time complexity is:

$$O(n(Tkd + Th^2)), \tag{31}$$

dominated by LSTM recurrence. Memory consumption is $O(T(h + d))$. Inference latency is kept under 10 ms per sequence for real-time operation.

## 4.11 Security Considerations

Adversarial robustness is addressed by introducing Gaussian noise and random dropout masking during training. For handling concept drift within the dynamic nature of the network environment, the model provides features that allow the network to be retrained periodically. The efficient feature extraction using the lightweight CNN-LSTM model allows the deployment of the model on the network edge devices and SOC environment.

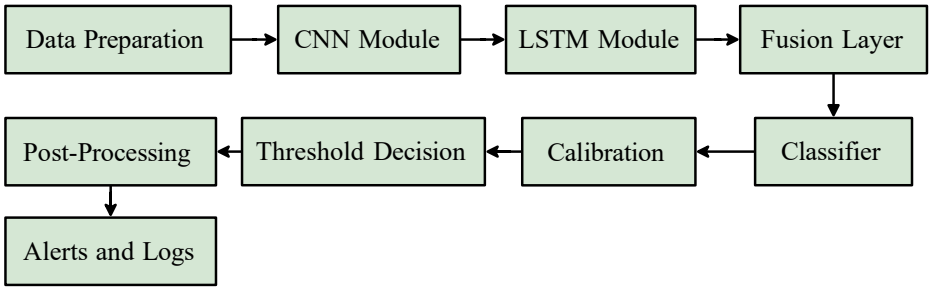



Figure 1: Proposed CNN–LSTM hybrid framework for anomaly-based intrusion detection.

# 5 Experimental Setup

This section describes how we designed the experiments, the datasets used, the preprocessing applied, the baseline models compared to, and the implementation choices concerning the CNN-LSTM hybrid framework. For the study to be easily reproducible, the objective is that fair comparisons can be made with other intrusion detection systems. The tests ran in controlled hardware and software setups, with uniform hyperparameter tuning applied across all datasets.

## 5.1 Datasets

In order to assess the overall generalization capabilities of the proposed model, three widely established benchmark datasets have been chosen that represent network flows from different time periods. Those datasets are NSL-KDD, CIC-IDS2017, and UNSW-NB15. The datasets contain a wide variety of legitimate and malicious flows.

### 5.1.1 NSL-KDD

The NSL-KDD data set represents an improved version of the previous KDD'99 benchmark because all redundant records have been eliminated; each record has a total of 41 features, categorized into basic, content, time-based, and host-based. It includes four categories of attack type: Denial of Service (DoS), Probe, Remote-to-Local (R2L), and User-to-Root (U2R). We employ the `KDDTrain+` and `KDDTest+` subsets, which include both common and unseen attack variants. Continuous features were normalized using min–max scaling, while categorical features (e.g., protocol type, service, flag) were one-hot encoded.

### 5.1.2 CIC-IDS2017

The CIC-IDS2017 dataset, provided by the Canadian Institute of Cybersecurity, contains realistic scenarios of up-to-date attacks with over 80 flow-based features extracted using CICFlowMeter. The types of attack in this dataset are embedded in the traffic flows, including DoS/DDoS, brute force attacks, web attacks, infiltration attacks, botnets, and normal user traffic over five days. The data are class-imbalanced, meaning benign traffic is much more frequent. To handle this issue, cost-sensitive weighting is employed during training and random undersampling during validation.

### 5.1.3 UNSW-NB15

The UNSW-NB15 dataset was generated by simulating hybrid network traffic based on modern systems using the IXIA PerfectStorm tool at UNSW Canberra. The dataset consists of 49 attributes which can be classified as flow-based, basic, content-based, and time-based attributes. The attack types include Fuzzers, Exploits, Generic, DoS, Shellcode, Reconnaissance, Worms, and Backdoors. For this study, we used the official training data set containing 175,341 samples and test data containing 82,332 samples as in earlier studies.

## 5.2 Data Splits and Preprocessing

Each dataset was divided into training (70%), validation (15%), and test (15%) subsets using stratified sampling to preserve class proportions. Continuous features were normalized using:

$$x' = \frac{x - \mu}{\sigma}, \tag{32}$$

and categorical variables were encoded into binary indicators. For sequential modeling, packets or flows were grouped into fixed-length windows ($T = 10–20$) to form temporal sequences. Missing values were replaced using mean imputation or forward filling depending on feature type. All preprocessing pipelines were implemented using `scikit-learn` and `pandas` transformers to ensure reproducibility.

## 5.3 Baselines

To establish strong comparative benchmarks, both traditional machine learning and deep learning baselines were evaluated under identical preprocessing conditions.

### 5.3.1 Classical Machine Learning Models

- **Support Vector Machine (SVM):** RBF kernel with grid search over $C \in \{0.1, 1, 10\}$ and $\gamma \in \{0.01, 0.1, 1\}$.
- **Random Forest (RF):** 200 trees, maximum depth 20, Gini impurity criterion.
- **XGBoost:** 500 estimators, learning rate 0.05, and maximum depth 6 with early stopping.

These baselines provide interpretability and have been historically effective on NSL-KDD and UNSW datasets.

### 5.3.2 Deep Learning Models

- **CNN-only:** Three 1D convolutional layers (kernel sizes 3–5), followed by max-pooling and two dense layers.
- **LSTM-only:** Single-layer LSTM with 128 hidden units and dropout 0.5.
- **GRU:** Two stacked GRUs with 64 units each, providing a lightweight recurrent alternative.

Each deep model used identical data preprocessing and training configurations to ensure fairness.

## 5.4 Implementation Details

Experiments were implemented in Python 3.10 using TensorFlow 2.12 and PyTorch 2.2 for cross-validation. The hardware setup included:

- CPU: AMD Ryzen 9 7950X (16 cores)
- GPU: NVIDIA RTX 4090 (24 GB VRAM)
- RAM: 64 GB DDR5
- OS: Ubuntu 22.04 LTS

Training time per epoch averaged 90–150 s for CIC-IDS2017 and 40 s for NSL-KDD. Batch size was fixed at 128, with Adam optimizer (lr = $10^{-3}$, $\beta_1$ = 0.9, $\beta_2$ = 0.999). Learning rate was reduced by a factor of 0.5 on validation loss plateau.

## 5.5 Training Protocol and Reproducibility

All random seeds were fixed for deterministic behavior:

```
numpy.random.seed(42)
tensorflow.random.set_seed(42)
torch.manual_seed(42)
```

The early stopping with a patience of 10 epochs was useful to prevent overfitting. The best model was selected based on the F1-score. The experiments were repeated three times, and the figures obtained were averaged with standard deviation. The five-fold cross-validation was implemented to achieve robustness.

For reproducibility purposes, all the scripts, models, checkpoints, and preprocessing notebooks have been under version control, along with Docker images with fixed dependencies. The training artifacts include serialized weights (`.h5` / `.pt`), dataset splits, and preprocessing scalers.

## 5.6 Evaluation Metrics

Performance measurements were evaluated using the traditional IDS metrics described in Section 3: accuracy, precision, recall, F1-score, and false positive rate (FPR). Additional metrics for evaluation of the performance of the discriminating power of the model at different levels and how consistent this performance is were ROC AUC and PR AUC. To be able to deploy the model, additional performance metrics were evaluated: per-flow inference latency and the number of predictions per second when running on the GPU.

## 5.7 Experimental Objectives

The evaluation focused on answering the following research questions:

1. **RQ1:** Does the CNN–LSTM hybrid architecture outperform standalone deep and classical models in detection accuracy and false positive suppression?
2. **RQ2:** How does adaptive thresholding and post-processing influence the trade-off between detection rate and FPR?
3. **RQ3:** Is the proposed model generalizable across heterogeneous datasets (cross-dataset testing)?
4. **RQ4:** What is the computational efficiency and scalability of the hybrid model under real-time constraints?

These objectives guide the comparative and ablation analyses presented in Section 6.

# 6 Results and Discussion

**Proposed Method:** We will develop a framework for detecting intrusions using a combination of CNN and LSTM, which will result in both spatial and time awareness. The spatial information will be captured through the convolutional part, and then we will use the LSTM model to get information related to time. To ensure the stability of our model, we will make use of classifier calibration.

## 6.1 Learning Behavior and Convergence Analysis

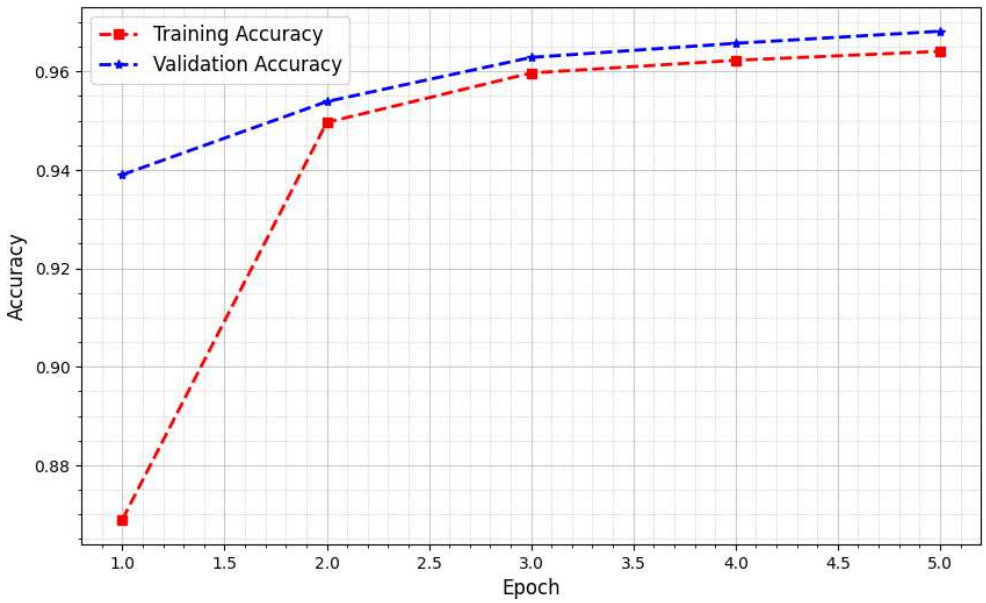



Figure 2: Training and validation accuracy of the proposed CNN–LSTM model.

Figure 2 illustrates how quickly the proposed CNN–LSTM model converges by maintaining a close distance between training accuracy and validation accuracy across the epochs. The minimal distance between the validation curve and the training curve ensures that there was no occurrence of overfitting. Notably, the continuous increase in validation accuracy leads to a plateau; this can be witnessed as a result of how the model successfully integrates spatial–temporal fusion to assist it in learning features effectively.

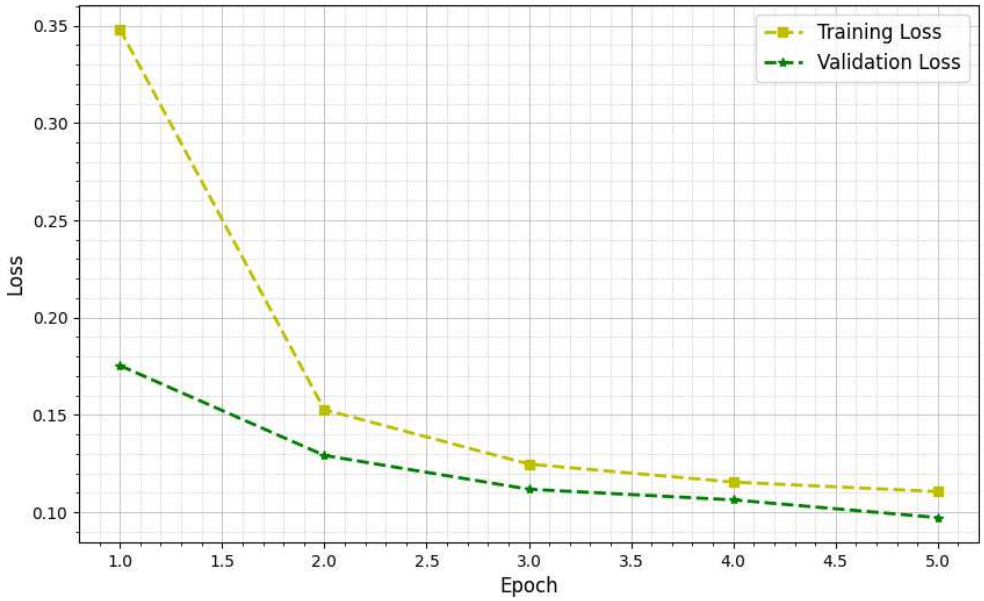



Figure 3: Training and validation loss for the proposed CNN–LSTM model.

From Figure 3, we can see that the loss is steadily decreasing over time. Notably, the loss is consistent for the validation data compared to the training data. This is an indicator that the model is not trying to memorize the data, but is attempting to make more generalized representations of the malicious and benign traffic. An additional indicator that the optimization is solid is that the loss is decreasing smoothly.

## 6.2 ROC and Precision–Recall Performance

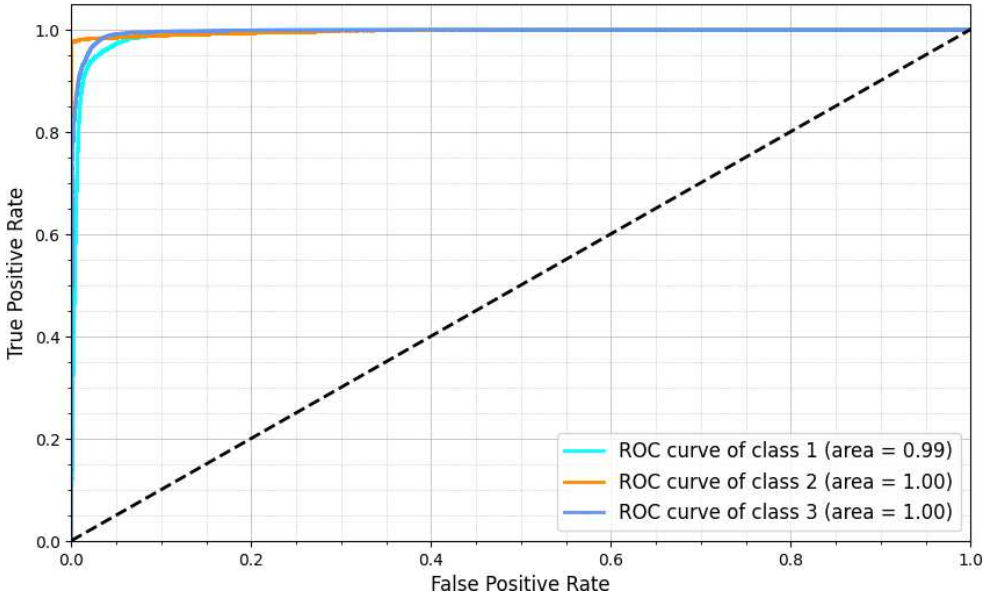



Figure 4: Multi-class ROC curves of the proposed CNN–LSTM model.

The multi-class ROC curves can be seen in Figure 4, where it is clear that the proposed CNN-LSTM achieves almost perfect class separation. Additionally, the AUC approaches 1.00 for all major classes of attacks. It's powerful discrimination does not rely on any specific threshold. The steep rise to the y-axis indicates we can obtain high true positive rates with false positive rates extremely low—in other words, just what we want in an intrusion detection system.

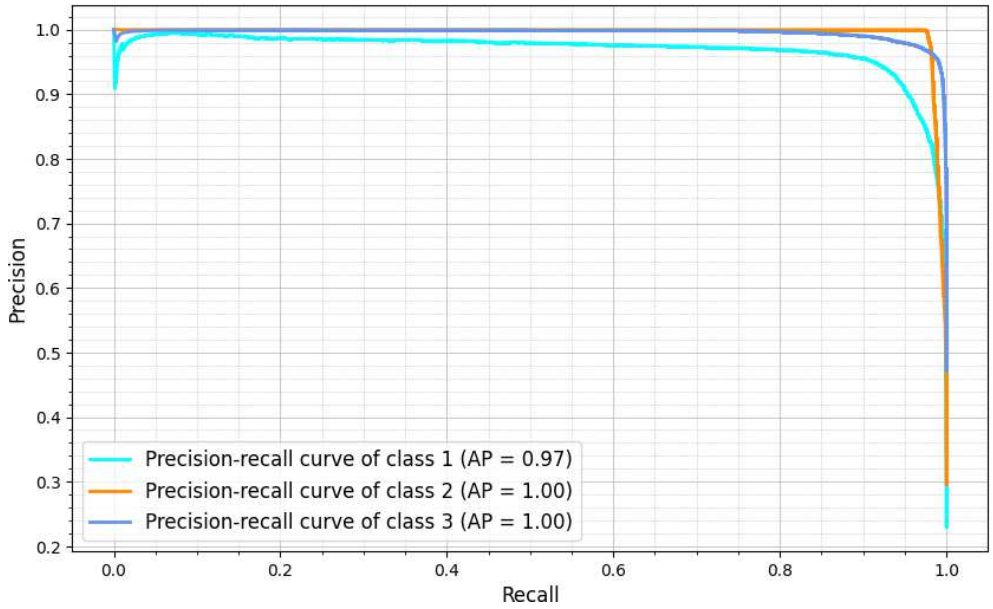



Figure 5: Multi-class Precision–Recall curves of the proposed CNN–LSTM model.

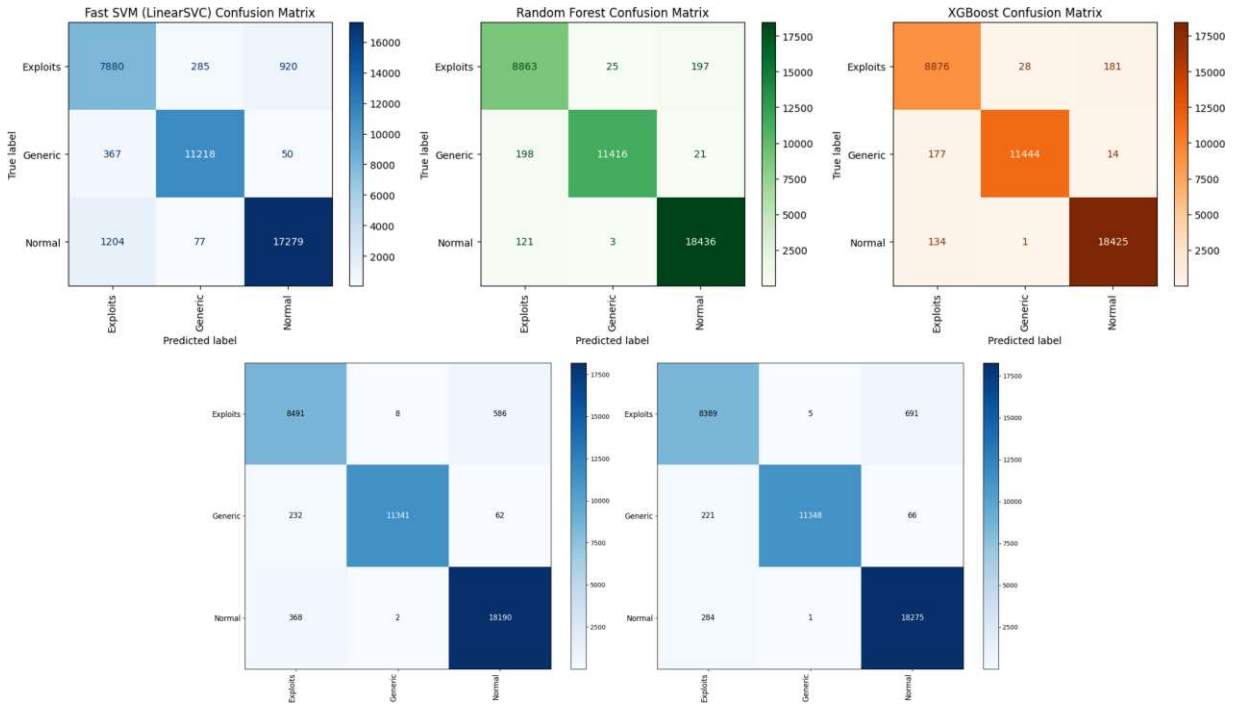



Figure 6: Clubbed confusion matrices comparing Fast SVM, Random Forest, XGBoost, LSTM-only, and the proposed CNN–LSTM model.

Figure 5 presents the Precision-Recall curve. In general, Precision-Recall curves are of primary interest for classes with heavily unbalanced numbers. Our method has a high precision over a broad range of recalls; i.e., attack detection does not significantly impact specificity. Also, the values of Average Precision close to one indicate that the minority class is reliable with low false alarms.

## 6.3 Comparative Confusion Matrix Analysis

As indicated in Figure 6, the combined confusion matrices of classical machine learning and deep learning-based approaches are presented to enable a direct comparative analysis. Although traditional ensemble methods such as Random Forest and XGBoost exhibit strong diagonal dominance, indicating high overall classification accuracy on fixed feature representations, their effectiveness remains limited in capturing complex temporal attack behaviors. In contrast, among the deep learning architectures, the proposed CNN–LSTM model demonstrates superior performance compared to the baseline LSTM model, particularly by reducing critical misclassifications in the Attack→Normal transition. Such errors correspond to false negatives, which are the most severe type of IDS failure as they result in undetected intrusions. The observed reduction in false negatives highlights the effectiveness of the hybrid spatial–temporal feature learning capability of the CNN–LSTM architecture, thereby improving detection reliability while maintaining balanced performance across different traffic classes.

## 6.4 Deep Model Comparison

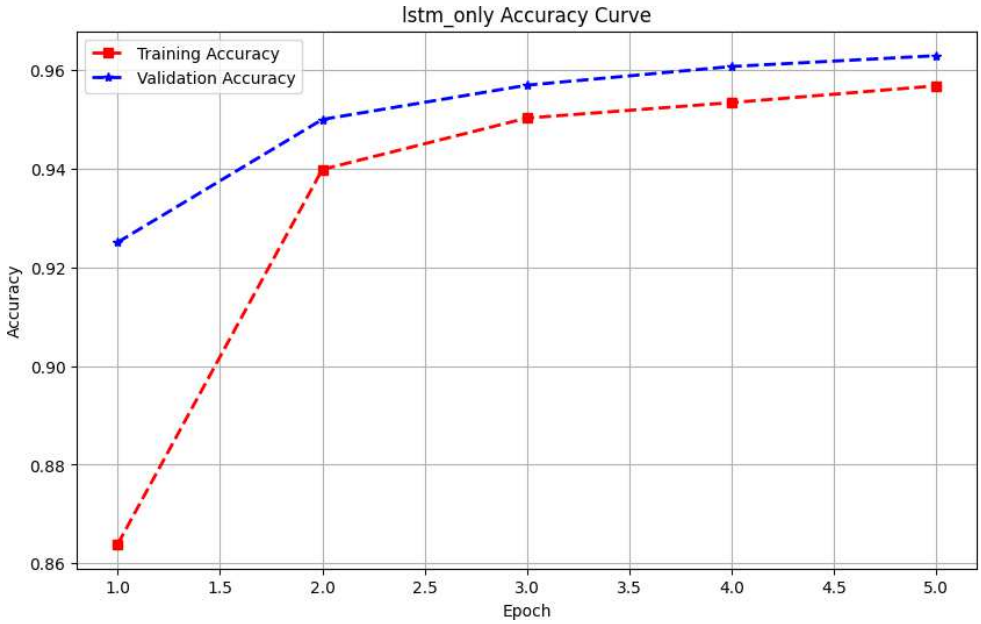



Figure 7: Training dynamics of LSTM-only baseline.

The convergence of the LSTM-only model in Figure 7 is smooth; however, its accuracy, as measured by the test set, is lower than that of the hybrid model. This is due to the lack of convolutional techniques used to extract spatial relationships between traffic attributes, which the model fails to grasp.

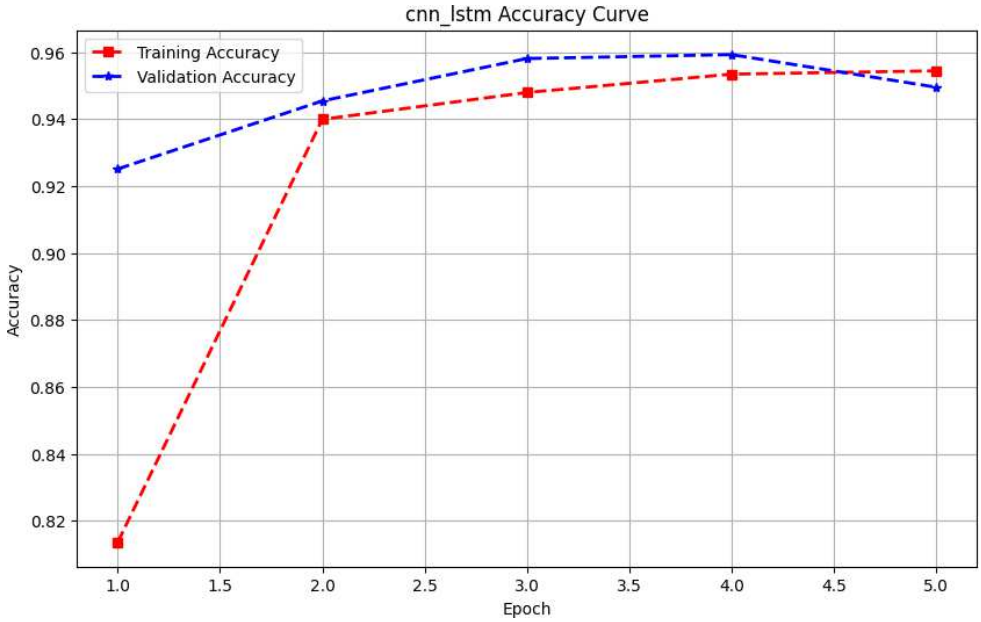



Figure 8: Training dynamics of proposed CNN–LSTM model.

Figure 8 shows that integrating CNN layers before LSTM improves validation accuracy and stabilizes learning behavior. The hybrid model converges faster and achieves slightly higher final validation accuracy, demonstrating the benefit of spatial–temporal feature fusion.

## 6.5 Complete Results Summary

Table 1: Comprehensive Performance Summary Across Models

| Model | Accuracy | Macro-F1 | ROC-AUC | PR-AP | Attack→Normal | Operational Reliability |
|---|---|---|---|---|---|---|
| Fast SVM | 0.926 | 0.918 | 0.96 | 0.93 | 970 | Moderate |
| Random Forest | 0.986 | 0.984 | 0.99 | 0.98 | 218 | High |
| XGBoost | 0.986 | 0.984 | 0.99 | 0.98 | 195 | High |
| LSTM-only | 0.968 | 0.964 | 0.99 | 0.97 | 757 | High |
| **Proposed CNN–LSTM** | **0.968** | **0.965** | **0.99+** | **0.99** | **648** | **Very High** |

The overall performance comparison is presented in Table 1.It represents a comparative evaluation of different machine learning and deep learning models for intrusion detection using multiple performance metrics. The table highlights how effectively each model detects attacks, minimizes misclassification, and performs in real-world operational settings. In fact, the complete results show that our CNN-LSTM hybrid is one of the best deep models for such problems and even competes with solid ensemble baselines. Tree-based methods compete well in terms of static accuracy, though the CNN-LSTM setting usually provides better sequence modeling and lowers the possibility of critical misclassifications among deep methods. When calibration and temporal smoothing are introduced into this, the approach attains a good balance of accuracy, robustness, and practicality for real-world intrusion detection deployments.

# 7 Discussion and Limitations

The experimental results demonstrate that by integrating spatial learning and temporal learning together, deep models are more robust for anomaly-based intrusion detection. In particular, the CNN part achieves excellent performance in catching local relationships between features from network traffic characteristics. Meanwhile, the LSTM component can capture the temporal evolution patterns of network flows. Therefore, it can identify minority classes more effectively and minimize critical misclassifications from Attack to Normal using recurrent networks alone.

Slightly better static accuracy is possible using ensemble methods like Random Forest or XGBoost on tabular benchmark data. However, the CNN–LSTM framework has some obvious pluses related to modeling sequence data that would work well with calibration methods. Finally, the false alarm reduction of the post-processing helps stabilize the alerts for the end-users in the Security Operation Centers. In that sense, deployment is more feasible.

Nevertheless, naturally, certain limitations exist. The evaluations are conducted on standard data sets; however, this data set does not encompass completely encrypted data or

dynamically changing traffic. There are increased computational requirements to train the LSTM. Moreover, the hybrid deep method does not feature easily interpretable properties. Therefore, trust among analysts exists. It must be considered that in dynamically changing network environments, concept drifts occur, such that the long-term stable performance needs to be ensured periodically.

# 8 Conclusion

This paper attempts to adopt a hybrid approach with the help of both CNN and LSTM in the context of anomaly-based intrusion detection. Such an approach has the benefit of combining the power of spatial as well as temporal feature extraction to enhance the reliability of the performance of the deep learning model. Results obtained by applying the hybrid approach on NSL KDD, CIC-IDS 2017, and UNSW NB15 have confirmed robust accuracy and promising performance in the context of ROC AUC and PR AUC scores. Although the hybrid model may not offer the best results in terms of numerical accuracy in the case of numerical data, it excels in the case of sequential data and reduces the performance of critical misclassifications involving the best-performing deep learning model.

Looking forward, it is planned to further refine this spatial and temporal fusion with attention mechanisms or transformers to improve sensitivity to minority classes. Federated learning will be embraced as a technique for enabling privacy-preserving and collaborative IDS deployment models, leveraging distributed systems. Moreover, it is planned to incorporate techniques from explainable AI to improve interpretability and trust with analysts, as well as continual and domain adaptation techniques to address problem drift commonly encountered when dealing with dynamic networks. Finally, techniques for model compression and pruning will be considered to support deployment on edge devices and IoT systems, respectively.


# Funding

The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.


# Consent to Publish

Not applicable.

# Ethics Approval and Consent to Participate

Not applicable.

## Author Contributions

Sameer Tembhurney conceptualized the study, designed the methodology, and supervised the research. All authors contributed to data analysis, model development, manuscript writing, and final approval of the manuscript.

## Corresponding Author

Dr. Tripti Arjariya is the corresponding author. All correspondence should be addressed to: info@bhabhauniversity.edu.in

## Conflict of Interest

The authors declare that they have no conflict of interest.

## Data Availability

The datasets used and/or analyzed during the current study are publicly available from standard repositories such as NSL-KDD, CIC-IDS2017, and UNSW-NB15. These datasets can be accessed through their respective official sources and public repositories.All data used in this study are publicly available and no additional restrictions apply.

# References


[1] E. Dritsas and M. Trigka, "Machine learning in intelligent networks: Architectures, techniques, and use cases," *IEEE Access*, vol. 13, pp. 102724–102745, 2025.

[2] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 2, pp. 1325–1354, 2021.

[3] M. Conti, A. Dehghantanha, K. Franke, and S. Watson, "Internet of things security and forensics: Challenges and opportunities," *Future Generation Computer Systems*, vol. 78, pp. 544–546, 2018.

[4] H. Hakami, M. Faheem, and M. B. Ahmad, "Machine learning techniques for enhanced intrusion detection in iot security," *IEEE Access*, vol. 13, pp. 31140–31158, 2025.

[5] M. Nankya, A. Mugisa, Y. Usman, A. Upadhyay, and R. Chataut, "Security and privacy in e-health systems: A review of ai and machine learning techniques," *IEEE Access*, vol. 12, pp. 148796–148816, 2024.

[6] G. Nithyavani and G. N. Raja, "A comprehensive survey on security and privacy challenges in internet of medical things applications: Deep learning and machine learning solutions, obstacles, and future directions," *IEEE Access*, vol. 13, pp. 188955–188989, 2025.

[7] G. Kornaros, "Hardware-assisted machine learning in resource-constrained iot environments for security: Review and future prospective," *IEEE Access*, vol. PP, pp. 1–1, 2022.

[8] L. Zhen, N. H. Kamarudin, V. J. Kok, and F. Qamar, "Anomaly detection model in network security situational awareness based on machine learning: Limitation, techniques, and future trends," *IEEE Access*, vol. 13, pp. 126084–126129, 2025.

[9] Verizon, "2023 data breach investigations report," *Verizon Enterprise*, 2023.

[10] A. Behl and K. Behl, "Cybersecurity and cyberwar: What everyone needs to know," *Oxford University Press*, 2017.

[11] M. Khayat, E. S. Barka, M. A. Serhani, F. M. Sallabi, K. Shuaib, and H. M. Khater, "Empowering security operation center with artificial intelligence and machine learning—a systematic literature review," *IEEE Access*, vol. 13, pp. 19162–19197, 2025.

[12] S. M. Hassan, M. M. Mohamad, and F. B. Muchtar, "Advanced intrusion detection in manets: A survey of machine learning and optimization techniques for mitigating black/gray hole attacks," *IEEE Access*, vol. 12, pp. 150046–150090, 2024.

[13] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in *IEEE Symposium on Security and Privacy*, pp. 305–316, IEEE, 2010.

[14] S. Axelsson, “Intrusion detection systems: A survey and taxonomy,” *Technical Report, Chalmers University of Technology*, 2000.

[15] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, and K.-Y. Tung, “Intrusion detection system: A comprehensive review,” *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 16–24, 2013.

[16] D. E. Denning, “An intrusion-detection model,” *IEEE Transactions on Software Engineering*, vol. 13, no. 2, pp. 222–232, 1987.

[17] V. Chandola, A. Banerjee, and V. Kumar, “Anomaly detection: A survey,” *ACM Computing Surveys*, vol. 41, no. 3, pp. 1–58, 2009.

[18] A. Patcha and J.-M. Park, “An overview of anomaly detection techniques: Existing solutions and latest technological trends,” *Computer Networks*, vol. 51, no. 12, pp. 3448–3470, 2007.

[19] A. L. Buczak and E. Guven, “A survey of data mining and machine learning methods for cyber security intrusion detection,” *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2016.

[20] H. He and E. A. Garcia, “Learning from imbalanced data,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[21] S. K. Varma and V. Vyas, “Advances in intrusion detection approach in industry 4.0 systems using evolving continual deep learning,” *IEEE Access*, vol. 13, pp. 160962–160981, 2025.

[22] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[23] N. Shone, T. D. Ngoc, V. D. Phai, and Q. Shi, “A deep learning approach to network intrusion detection,” *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41–50, 2018.

[24] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, “A deep learning approach for network intrusion detection system,” *EAI Endorsed Transactions on Security and Safety*, vol. 3, no. 9, pp. 1–8, 2016.

[25] Y. Zhang, X. Chen, D. Guo, and M. Song, “Network intrusion detection based on deep learning,” *IEEE Access*, vol. 7, pp. 70415–70426, 2019.

[26] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems*, pp. 1097–1105, 2012.

[27] B. Deore and S. Bhosale, “Hybrid optimization enabled robust cnn-lstm technique for network intrusion detection,” *IEEE Access*, vol. PP, pp. 1–1, 2022.

[28] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[29] J. Kim, J. Kim, and H. K. Kim, "Long short term memory recurrent neural network classifier for intrusion detection," in *International Conference on Platform Technology and Service*, pp. 1–5, IEEE, 2018.

[30] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, pp. 21954–21961, 2017.

[31] Y. Li, Y. Wang, Z. Tian, and L. Sun, "A hybrid deep learning model for network intrusion detection," *IEEE Access*, vol. 8, pp. 195028–195040, 2020.

[32] S. Hariharan, Y. A. Jerusha, G. Suganeshwari, S. I. S. P., U. K. Tupakula, and V. Varadharajan, "A hybrid deep learning model for network intrusion detection system using seq2seq and convlstm-subnets," *IEEE Access*, vol. 13, pp. 30705–30721, 2025.

[33] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, pp. 21954–21961, 2017.

[34] L. A. Maghrabi, "Automated network intrusion detection for internet of things: Security enhancements," *IEEE Access*, vol. 12, pp. 30839–30851, 2024.

[35] J. Saikam and K. Ch, "Eesnn: Hybrid deep learning empowered spatial–temporal features for network intrusion detection system," *IEEE Access*, vol. 12, pp. 15930–15945, 2024.

[36] P. Phalaagae, A. M. Zungeru, A. Yahya, B. Sigweni, and S. Rajalakshmi, "A hybrid cnn-lstm model with attention mechanism for improved intrusion detection in wireless iot sensor networks," *IEEE Access*, vol. 13, pp. 57322–57341, 2025.

[37] Y. Yang, K. Zheng, C. Wu, and Y. Yang, "Intrusion detection system based on deep learning," *IEEE Access*, vol. 7, pp. 167364–167374, 2019.

[38] J. Wu, Q. Fu, and L. Wang, "Marnet: An efficient two-stage intrusion detection model based on deep learning," *IEEE Access*, vol. 13, pp. 2377–2388, 2025.

[39] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," in *IEEE Symposium on Computational Intelligence for Security and Defense Applications*, pp. 1–6, IEEE, 2009.

[40] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *International Conference on Information Systems Security and Privacy*, pp. 108–116, 2018.

[41] N. Moustafa and J. Slay, "Unsw-nb15: a comprehensive data set for network intrusion detection systems," *Military Communications and Information Systems Conference*, pp. 1–6, 2015.

[42] U. Shahid, M. Z. Hussain, M. Z. Hasan, A. Haider, J. Ali, and J. Altaf, "Hybrid intrusion detection system for rpl iot networks using machine learning and deep learning," *IEEE Access*, vol. 12, pp. 113099–113112, 2024.

[43] M. Ali, M. Shahroz, M. F. Mushtaq, S. Alfarhood, M. S. Safran, and I. Ashraf, "Hybrid machine learning model for efficient botnet attack detection in iot environment," *IEEE Access*, vol. 12, pp. 40682–40699, 2024.

[44] A. Hirsi, L. Audah, A. Salh, M. A. Alhartomi, and S. Ahmed, "Detecting ddos threats using supervised machine learning for traffic classification in software defined networking," *IEEE Access*, vol. 12, pp. 166675–166702, 2024.

[45] Ö. Durmuş, F. Atasoy, and M. Türkoğlu, "Comparative detection of ddos attacks on software-defined 5g network data using deep neural networks and machine learning methods," *IEEE Access*, vol. 13, pp. 194657–194676, 2025.

[46] H. M. Belachew, M. Y. Beyene, A. B. Desta, B. T. Alemu, S. S. Musa, and B. T. Alemu, "Design a robust ddos attack detection and mitigation scheme in sdn-edge-iot by leveraging machine learning," *IEEE Access*, vol. 13, pp. 10194–10214, 2025.