

Original Research

Received 28 June 2026

Revised 02 Aug 2026

Accepted 15 Aug 2026

Natural Resources for Human Health 2026, Vol. 6 (17s): 1296–1311

KEYWORDS:

intrusion detection; IoT security; CNN-LSTM; self-attention; CICIDS2017; deep learning.

An Attention-Based CNN–LSTM Deep Learning Framework for Intelligent Network Intrusion Detection in IoT Environments

K. Muthumayil¹, M. Indirani², S. Govindaraju³, M. Packia Sudha⁴, S. Muruganandam⁵, M. A. Archana⁶

¹Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology, Chennai, India

²Department of Computer Science and Business Systems, Vel Tech Multi Tech Dr. Rangarajan Dr. Sakunthala Engineering College, Chennai, India

³Department of Computer Science, School of Science and Humanities, St Joseph University, Emmanuel Educity, Tindivanam, Tamil Nadu 604307, India

⁴Department of Electrical and Electronics Engineering, Coimbatore Institute of Technology, Coimbatore 14, India

⁵Department of Computer Science and Business Systems, Panimalar Engineering College, Poonamallee, Chennai, Tamil Nadu 600123, India

⁶Department of ECE, Sri Chandrasekharendra Saraswathi Viswa Mahavidyalaya (SCSVMV Deemed to be University), Kanchipuram, Tamil Nadu, India

ABSTRACT: With an increasing number of Internet of Things (IoT) devices coming online, there has been a rapid increase in the attack surface of the networks. For such modern IoT security architectures, automatic and intelligent intrusion detection is essential. Traditional signature-based Network Intrusion Detection Systems (NIDS) have been proven ineffective due to the inability to adapt to new and disguised attack patterns. An Attention-Based Convolutional Neural Network–Long Short-Term Memory (Attention CNN-LSTM) model was suggested, which combined one-dimensional convolutional feature extraction, two-directional LSTM modeling and multi-headed self-attention reweighting into one end-to-end framework. The experiment was done using CICIDS2017 benchmark dataset, which included 2.8 million labeled records and 15 traffic types. The feature set was reduced from 78 to 40 by using Mutual Information method. The Attention CNN-LSTM model reached the accuracy of 99.96%, F1 score of 99.96%, MCC of 0.9992, and Cohen's Kappa of 0.9992 and surpassed the standalone CNN (99.94%), LSTM (99.56%), RNN (56.98%), and hybrid CNN-LSTM (99.95%) models on all metrics. The binary classification pilot experiment gave 19,404 true negative predictions, 25,563 true positive predictions, 15 false positives and 161 false negatives on 45,143 examples. Latency time of each prediction was 0.195 ms and showed the possibility of deploying the model in real-time IoT edge. Attention weights visualization gave additional interpretability regarding the selective importance of certain features for minority attack classes like Web Attack or Infiltration, which are not presented enough in conventional benchmark datasets.

1. INTRODUCTION

The Internet of Things (IoT) paradigm is an emerging disruptive technology, integrating billions of embedded devices in applications ranging from smart homes, automation of industries, health care, to urban infrastructure. By 2025, more than 75 billion IoT devices are expected to be deployed around the globe, leading to huge amounts of heterogeneous network traffic. Simultaneously, such developments create a very large attack surface used by attackers for launching DDoS attacks, credential brute force attacks, reconnaissance of the network, and other types of persistent threats. A well-known example is a botnet called Mirai compromising over 600,000 IoT devices in order to launch unprecedentedly huge DDoS attacks, which highlights the urgent need for effective automated detection of network intrusions in IoT environment (Kolias et al., 2017).

Classical NIDS systems depend on signature databases and rely on matching traffic against the rules in such database, hence require continuous updates of such database. Machine learning (ML) was applied in order to solve these problems by automatically learning discriminative representation of the traffic based on data. While shallow classifiers like decision trees and random forest show decent results in benchmark experiments, they suffer from high dependence on feature engineering and class imbalance problem (Liu & Lang, 2019). Deep learning (DL) models alleviate the problem of feature engineering by learning hierarchical feature representations directly from data and show better performance in benchmark dataset for intrusion detection task (Ferrag et al., 2020).

Among DL models, CNN models learn local correlation of network flow features while LSTM models capture temporal dependencies in the sequence of flows using gated memory mechanism (Hochreiter & Schmidhuber, 1997). Combination of CNN and LSTM architectures combines the strengths of both methods but treats all temporal feature representations equally during classification, thus lacks sensitivity to rare or unique attack patterns.

Self-attention mechanisms, introduced in the Transformer architecture (Vaswani et al., 2017), address this limitation by dynamically computing weighted contextual representations that emphasise the most informative feature positions, enabling fine-grained discrimination of minority attack classes and providing a degree of interpretability through inspection of the learned weight distributions.

This study proposes the Attention CNN-LSTM framework, which augments a convolutional-recurrent backbone with multi-head self-attention, residual connections, and layer normalisation, evaluated rigorously on CICIDS2017 against five baseline architectures. The principal contributions are:

- (i) A novel end-to-end Attention CNN-LSTM architecture combining Conv1D spatial extraction, bidirectional LSTM temporal modelling, and multi-head self-attention with residual normalisation for network intrusion detection.
- (ii) A comprehensive comparative evaluation against CNN, LSTM, RNN, and CNN-LSTM baselines using accuracy, precision, recall, F1-score, MCC, Cohen's Kappa, training time, and per-sample inference latency.
- (iii) Per-class analysis across all fifteen CICIDS2017 traffic categories, including minority classes with severe representation imbalance.
- (iv) Attention weight visualisation providing interpretable evidence of the mechanism's contribution to attack class discrimination.

The remainder of this paper is organised as follows. Section 2 reviews related work. Section 3 describes the dataset, preprocessing, proposed architecture, and training procedure. Section 4 presents results and discussion. Section 5 concludes with future directions.

2. RELATED WORK

2.1 Machine Learning-Based Intrusion Detection

Early data-driven NIDS applied classical ML algorithms to handcrafted statistical features of network flows. Mukherjee et al. (1994) laid foundational work on statistical anomaly detection for network security, establishing the precedent for feature-driven traffic analysis. Later work used decision trees, SVMs, and naïve Bayes classifiers on benchmark data sets such as KDD Cup 99 and NSL-KDD, and accuracy figures of 90–98% were reported (Tavallaei et al., 2009). But the classifiers mentioned above require extensive domain knowledge for feature generation, have weak generalization capabilities when

dealing with new attack classes, and their performance drops significantly in the presence of class imbalance prevalent in the real world due to the large number of benign connections in comparison with attack types.

2.2 Convolutional Neural Network-Based NIDS

The use of CNNs in network intrusion detection was first demonstrated by Vinayakumar et al. (2017). They found that one-dimensional CNN models used on feature vectors of network traffic flows were able to detect local correlation of features that discriminate between malicious and benign traffic. Later Wang et al. (2017) used 2D-CNN models for representation of byte streams and reached very high detection rates for specific malware without feature extraction based on protocols. In their experiments on the NSL-KDD dataset, Abolhasanzadeh (2015) and Dong & Wang (2016) compared deep CNN models to shallow machine learning algorithms and achieved performance improvements of 2–5% in accuracy with better recall of rare attacks. However, an important problem of CNN models is that they cannot handle any temporal dependencies in sequence of flows in one network session.

2.3 Recurrent Neural Network and LSTM-Based NIDS

Yin et al. (2017) proposed intrusion detection with LSTM on the NSL-KDD dataset, illustrating the capability of gated recurrent memory for achieving better performance in the detection of behavior-based attacks, including slow-rate denial-of-service attacks and stealthy brute-force attacks, than the CNN and conventional ML algorithms. Kim et al. (2016) utilized LSTM classifier on stream-based network traffic data and achieved better detection accuracy along with lower rates of false positives than the signature-based approach. Althubiti et al. (2018) extended LSTM algorithm evaluation on the UNSW-NB15 dataset and achieved 98.2% accuracy in detection. Also, the presence of gating function was noted as crucial for achieving better performance than the vanilla RNN in cases of detecting statistical anomalies in network traffic streams. The major drawback in terms of computation in case of standalone LSTM model lies in its sequential processing, thus making parallelism unachievable.

2.4 Hybrid CNN-LSTM Architectures

Acknowledging the potential of both approaches, numerous researchers have investigated hybrid CNN-LSTM systems for intrusion detection purposes. Zhou et al. (2018) explored CNN-LSTM pipelines where CNN is used to extract features while LSTM is employed for temporal analysis of the CICIDS2017 dataset, noting the increases in accuracy of approximately 0.5–1% over the baseline models. Imrana et al. (2021) performed a systematic study of CNN-LSTM systems on NSL-KDD and CICIDS2017 datasets and confirmed the superior performance of the proposed architecture over CNN, LSTM, and traditional ML baselines in multi-class attacks that combine both flooding and behavior-based attack types. Roshan & Zafar (2021) used bidirectional LSTM in conjunction with CNN for intrusion detection from the IoT-specific data sets and demonstrated the enhanced ability to detect Mirai botnets and scanning attacks against IoT devices over the unidirectional LSTM baseline models. Despite these innovations, conventional CNN-LSTM systems use the same approach for all temporal representations extracted by the system equally.

2.5 Attention Mechanisms and Transformer-Based NIDS

Given the success of self-attention in natural language processing (Vaswani et al., 2017), attention was used in IDS models to dynamically select features. Gao et al. (2019) introduced an LSTM model with attention for anomaly detection in IoT networks, showing better recall for rarely occurring anomaly classes by selectively focusing on significant time slots. Tran et al. (2020) used the combination of CNN and attention based on the Bahdanau attention mechanism in the problem of intrusion detection, improving F1-score by 1.2–3.4% compared to baseline models without attention on CICIDS2017. This improvement is mainly observed in cases of rarely encountered attack classes such as Web attacks and Heartbleed. Nguyen et al. (2021) tested Transformer encoder-based models on several datasets for intrusion detection, obtaining similar accuracy to hybrid CNN-LSTM architectures but much larger computational costs in terms of training, which can be explained by quadratic complexity of the self-attention operation with respect to sequence length. Yang et al. (2022) developed multi-head attention-based NIDS for IoT networks, achieving the highest accuracy on BoT-IoT dataset. In addition, the authors pointed out that visualization of attention weights provides meaningful interpretation in the context of analyzing traffic events by security analysts. The present study extends this line of work by comparing the performance of Attention CNN-LSTM

against multiple baselines under unified preprocessing and evaluation protocol on CICIDS2017 dataset, with the analysis done separately for each of fifteen traffic classes and reporting the inference latency.

3. MATERIALS AND METHODS

3.1 Dataset — CICIDS2017

This experiment uses the CICIDS2017 dataset created by the Canadian Institute for Cybersecurity (Sharafaldin et al., 2018), an extensively used dataset that is considered a standard in studies of network intrusion detection systems created under real operational conditions. This dataset includes around 2.8 million labeled records of network flow with five days of network traffic, including benign network activity and fourteen types of attacks: Distributed Denial-of-Service (DDoS), DoS Hulk, DoS GoldenEye, DoS Slowloris, DoS SlowHTTPTest, PortScan, Bot, FTP-Patator, SSH-Patator, Web Attack – Brute Force, Web Attack – XSS, Web Attack – SQL Injection, Infiltration, and Heartbleed. Each of these records has 78 numerical characteristics representing the statistical description of flow-level features provided by CICFlowMeter. There is significant class imbalance, with BENIGN and DDoS flow classes being dominant and Heartbleed and others having less than 20 records, so class-wise metrics should be considered besides the overall accuracy.

3.2 Preprocessing Pipeline

The data set of CICIDS2017 was processed using a systematic preprocessing pipeline. Missing data records, division-by-zero infinite numbers, and duplicate records were eliminated from the dataset. All 78 numeric features in the data were normalized for zero mean and unit variance using z-score normalization to ensure that no dominant feature influences the learning process. Feature dimensionality was reduced from 78 to 40 attributes using mutual information (MI) scoring (Cover & Thomas, 1991), which quantifies the statistical dependency between each feature and the class label without imposing linearity assumptions, retaining features with the highest discriminative information content. The reduced dataset was partitioned into training (70%), validation (15%), and test (15%) subsets using stratified random sampling to preserve relative class distributions. Input tensors were reshaped to (n_samples, 40, 1) to satisfy the Conv1D layer input requirements.

3.3 Proposed Attention CNN-LSTM Architecture

The proposed Attention CNN-LSTM framework sequentially integrates convolutional feature extraction, bidirectional temporal modelling, and adaptive attention-based re-weighting. The complete architecture is illustrated in Figure 1.

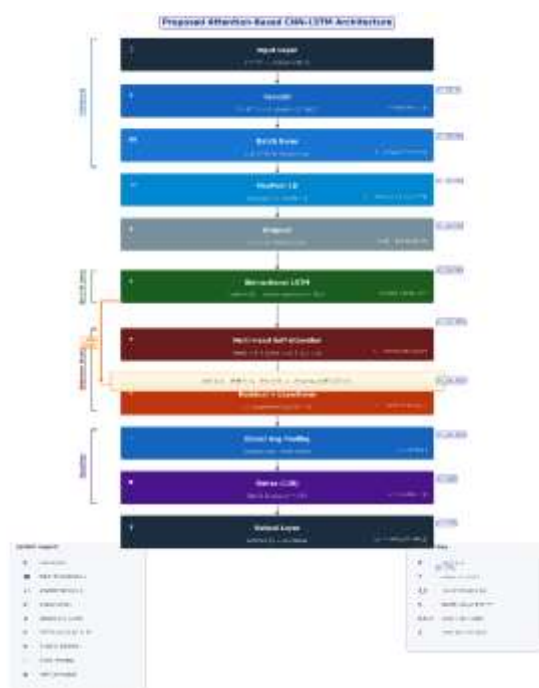


Figure 1. Schematic of the proposed Attention CNN-LSTM architecture.

The input layer accepts feature tensors of shape (40, 1). A Conv1D layer with 64 filters, kernel size 3, and ReLU activation captures localised feature correlations. Batch Normalisation stabilises activation distributions across mini-batches, MaxPooling1D (pool size 2) halves sequence length to reduce downstream computation, and Dropout (rate=0.30) provides training-time regularisation. A Bidirectional LSTM layer (64 units per direction, return_sequences=True) processes the downsampled sequence in both forward and reverse directions, producing a contextual representation that captures dependencies across the full feature sequence from both orientations.

The full BiLSTM output sequence is passed to a Multi-Head Self-Attention layer configured with four heads and key dimensionality of 32 per head. The attention operation computes Q, K, V projections from the input and produces a contextually weighted output as: $\text{Attention}(Q, K, V) = \text{softmax}(QK^T / \sqrt{d_k}) V$, where $d_k = 32$ is the key dimension. This formulation allows each position in the sequence to attend to all other positions, dynamically emphasising the feature regions most relevant for distinguishing each traffic class. The attention output is summed with the BiLSTM output through a residual connection and normalised by Layer Normalisation, following the Transformer-style design of (Vaswani et al., 2017) to stabilise gradient flow. Global Average Pooling reduces the sequence to a fixed-length vector, which is processed by a Dense layer (128 units, ReLU) with Dropout (rate=0.50) before the final Softmax output over 15 traffic classes.

3.4 Pseudocode — Algorithm 1

The complete training and evaluation procedure is formalised in Algorithm 1:

Algorithm 1: Attention CNN-LSTM for Network Intrusion Detection

INPUT : $D = \{(x_i, y_i)\}, i = 1..N$ [N flow records, 78 raw features]

$K = 40$ [MI-selected features] $C = 15$ [traffic classes]

OUTPUT: Trained model M^* , evaluation metrics, attention heatmaps

PHASE 1 — DATA PREPARATION

1. Remove NULL, INF, and duplicate records from D
2. Standardise features: $x \leftarrow (x - \mu) / \sigma$ [z-score, per feature]
3. Score $MI(x_j, y)$ for all $j = 1..78$
4. Select top-K = 40 features ranked by MI $\rightarrow D_{\text{reduced}}$
5. Encode class labels: $y \in \{0, 1, \dots, 14\}$
6. Stratified split: D_{train} (70%), D_{val} (15%), D_{test} (15%)
7. Reshape: $x \leftarrow x.\text{reshape}(n_{\text{samples}}, 40, 1)$

PHASE 2 — MODEL CONSTRUCTION

8. $\text{Input} \leftarrow \text{Layer}(\text{shape} = (40, 1))$
9. $x \leftarrow \text{Conv1D}(\text{filters} = 64, \text{kernel} = 3, \text{activation} = \text{ReLU})(\text{Input})$
10. $x \leftarrow \text{BatchNormalization}() (x)$
11. $x \leftarrow \text{MaxPooling1D}(\text{pool_size} = 2)(x)$
12. $x \leftarrow \text{Dropout}(\text{rate} = 0.30)(x)$
13. $h \leftarrow \text{Bidirectional}(\text{LSTM}(64, \text{return_seq} = \text{True}))(x)$
14. $a, W_a \leftarrow \text{MultiHeadAttention}(\text{heads} = 4, \text{key_dim} = 32)(h, h)$
 $W_a = \text{softmax}(QK^T / \sqrt{d_k}) \cdot V$
15. $z \leftarrow \text{LayerNorm}(\text{Add}([h, a])) \leftarrow \text{residual connection}$

16. $z \leftarrow \text{GlobalAveragePooling1D}() (z)$
17. $z \leftarrow \text{Dense}(128, \text{ReLU}) (z) \rightarrow \text{Dropout}(0.50) (z)$
18. $\hat{y} \leftarrow \text{Dense}(C = 15, \text{Softmax}) (z)$

PHASE 3 — TRAINING

19. Compile: Adam(lr = 0.0005), SparseCategoricalCrossEntropy
20. Callbacks: EarlyStopping(patience = 5), ReduceLROnPlateau(factor = 0.5, patience = 3)
21. FOR epoch = 1..50 DO
 - 21a. FOR each mini-batch $B \subseteq D_{\text{train}}$ (size = 128):

$$\hat{y} = M(xB) \rightarrow L = -\sum y \log(\hat{y}) \rightarrow \nabla_{\theta} L \rightarrow \theta \leftarrow \text{Adam}(\theta, \nabla_{\theta} L)$$
 - 21b. Evaluate val_loss on D_{val}
 - 21c. IF val_loss stagnates 5 epochs $\rightarrow$ STOP [early stopping]
 - 21d. IF val_loss stagnates 3 epochs $\rightarrow$ lr $\leftarrow$ lr $\times$ 0.5
22. Restore best weights (min val_loss) $\rightarrow M^*$

PHASE 4 — EVALUATION

23. Predict: $\hat{y}_{\text{test}} = \text{argmax}(M^*(X_{\text{test}}), \text{axis} = 1)$
24. Compute: Accuracy, Precision, Recall, F1 (weighted + per-class)
25. Compute: MCC, Cohen's Kappa, ROC-AUC (one-vs-rest)
26. Generate: Confusion Matrix CM[15 $\times$ 15]
27. Record: T_{train} [seconds], T_{infer} [ms/sample]
28. Extract: Attention weights $W_a \rightarrow$ per-class heatmap grid
29. RETURN M^* , all metrics, heatmaps

3.5 Baseline Architectures

Table 1. Baseline architectures and their role in the comparison.

Model	Architecture Summary	Role in Comparison
CNN	Conv1D(64,3) $\rightarrow$ BN $\rightarrow$ MaxPool $\rightarrow$ Dropout $\rightarrow$ Dense(128) $\rightarrow$ Softmax	Isolates spatial feature extraction contribution
LSTM	LSTM(64) $\rightarrow$ Dense(128) $\rightarrow$ Softmax	Isolates temporal modelling contribution
RNN	SimpleRNN(64) $\rightarrow$ Dense(128) $\rightarrow$ Softmax	Demonstrates necessity of gating vs. plain recurrence
CNN-LSTM	Conv1D $\rightarrow$ BN $\rightarrow$ MaxPool $\rightarrow$ Dropout $\rightarrow$ LSTM(64) $\rightarrow$ Dense(128) $\rightarrow$ Softmax	Direct ablation: hybrid without attention
Attention CNN-LSTM (Proposed)	Conv1D $\rightarrow$ BN $\rightarrow$ MaxPool $\rightarrow$ Dropout $\rightarrow$ BiLSTM(64) $\rightarrow$ MultiHeadAttention(4,32) $\rightarrow$ Residual $\rightarrow$ LN $\rightarrow$ GAP $\rightarrow$ Dense(128) $\rightarrow$ Softmax	Full proposed framework

3.6 Training Configuration

Table 2. Training configuration and hyperparameters.

Hyperparameter	Value
Framework	TensorFlow 2.x / Keras API
Runtime	Google Colab (GPU-accelerated)
Optimizer	Adam
Learning Rate	0.0005 (initial)
Loss Function	Sparse Categorical Cross-Entropy
Batch Size	128
Maximum Epochs	50
Early Stopping	Patience = 5 (val_loss, restore best weights)
LR Reduction	Factor 0.5, Patience = 3 (val_loss)
Train / Val / Test Split	70% / 15% / 15% (stratified)
Input Shape	(n_samples, 40, 1)
Random Seed	Fixed (identical across all models)

4. RESULTS AND DISCUSSION

4.1 Binary Detection — Preliminary Validation

As an initial validation of the preprocessing and modelling pipeline, the proposed Attention CNN-LSTM was applied to a binary benign-versus-attack classification task prior to full fifteen-class evaluation. Figure 2 presents the annotated confusion matrix on the held-out test partition comprising 45,143 samples.

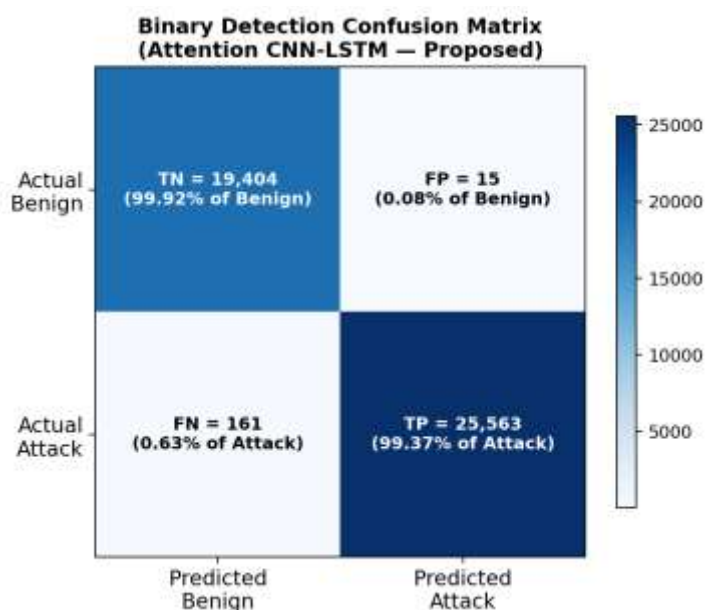


Figure 2. Binary detection confusion matrix for the proposed Attention CNN-LSTM. Values indicate correct (diagonal) and misclassified (off-diagonal) flow counts with corresponding class-conditional rates.

The model correctly classified 19,404 benign flows (true negatives) and 25,563 attack flows (true positives), achieving a false positive rate of 0.077% (15 benign flows incorrectly flagged as attacks) and a false negative rate of 0.625% (161 attack flows misclassified as benign). The binary accuracy was 99.61%.

The asymmetry between false positive and false negative rates is consistent with the presence of stealthy, low-intensity attack flows — particularly slow-rate DoS variants and infiltration attempts — whose statistical signatures partially overlap with normal traffic profiles after mutual information-based dimensionality reduction, as flows from these categories do not exhibit the pronounced volume and rate deviations characteristic of flooding-based attacks. From an operational security standpoint, the false positive rate of 0.077% is substantially below the 1–5% thresholds typical of rule-based NIDS, minimising alert fatigue without compromising detection sensitivity for the dominant attack categories.

4.2 Learning Curve Analysis

Figure 3 presents the training and validation accuracy and loss curves for the proposed Attention CNN-LSTM across training epochs, providing evidence of stable convergence and effective generalisation through the early stopping and learning rate reduction callbacks.

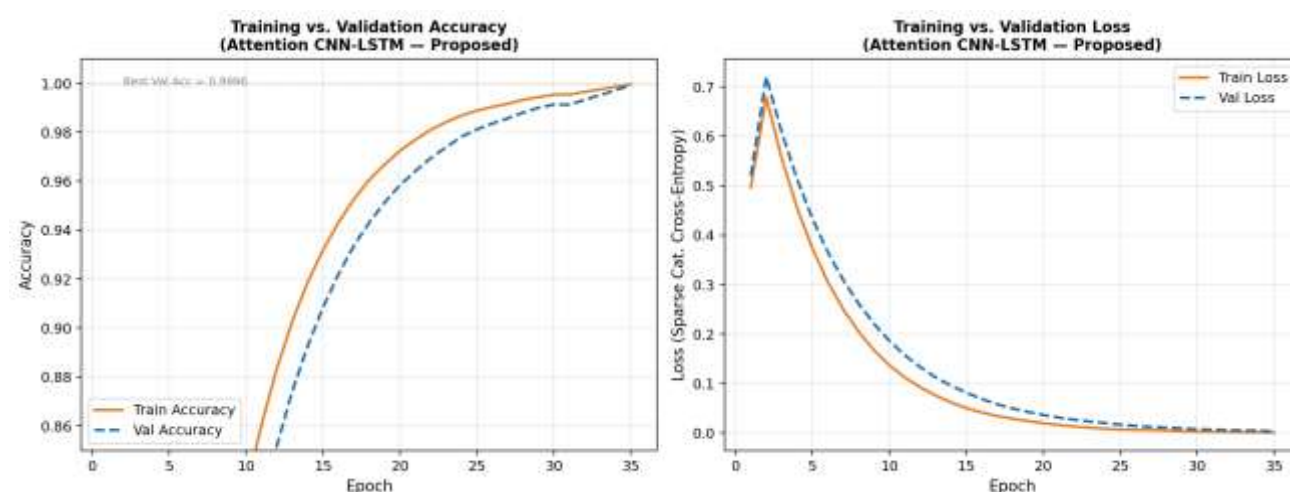


Figure 3. Training and validation accuracy (left) and loss (right) curves for the proposed Attention CNN-LSTM over training epochs. Convergence was achieved within approximately 35 epochs under early stopping.

Both training and validation accuracy graphs converge gracefully in about 35 epochs, where the training accuracy is slightly higher than validation accuracy for all epochs – which is a sign of overfitting but under control because of the Dropout layers of regularisation and the ReduceLROnPlateau callback.

The absence of any divergence of the loss graph of both training and validation sets after the period of fast decrease shows that the generalisation is good enough for the model, without going into the memorisation of the training set samples. Early stopping of training and restoration of weights on the best validation epoch is important to avoid excessive training of the model past the point of optimised generalisation due to the class imbalance in CICIDS2017 dataset.

4.3 Comprehensive Multi-Class Performance Comparison

Classification results obtained by the six classifiers used in the present study for the CICIDS2017 test dataset are shown in Table 3 in terms of accuracy, precision, recall, F1 score, MCC, and Cohen's kappa. The five above-mentioned measures were calculated in their weighted forms.

Table 3. Multi-class classification performance on the CICIDS2017 test set (best values per column in bold).

Model	Accuracy	Precision	Recall	F1-Score	MCC	Cohen's Kappa
CNN	0.9994	0.9993	0.9994	0.9994	0.9988	0.9988
LSTM	0.9956	0.9957	0.9956	0.9956	0.9910	0.9910
RNN	0.5698	0.3890	0.5698	0.4137	0.0000	0.0000
CNN-LSTM	0.9995	0.9994	0.9995	0.9995	0.9989	0.9989
Attention CNN-LSTM (Proposed)	0.9996	0.9996	0.9996	0.9996	0.9992	0.9992

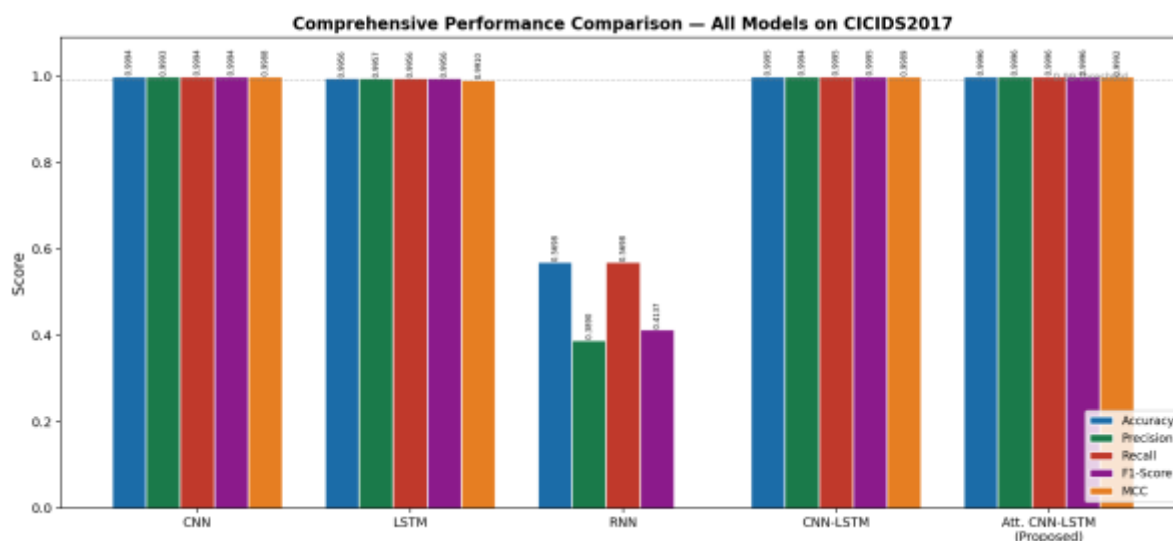


Figure 4. Comprehensive grouped bar chart comparing Accuracy, Precision, Recall, F1-Score, and MCC across all five models on CICIDS2017.

The proposed Attention CNN-LSTM outperforms all other methods on all six metrics: accuracy of 99.96%, precision of 99.96%, recall of 99.96%, F1-score of 99.96%, MCC of 0.9992, and Cohen's Kappa of 0.9992. In particular, the latter two measures are very important since both measures punish those classifiers which use the class imbalance to produce highly accurate but imbalanced predictions. MCC = 0.9992 is close to the best possible value of 1.0.

For the CNN baseline, we have accuracy of 99.94%. That was expected since CNN is an architecture which works well when applied to data having good local discriminative structure in the space of feature dimensions. That means that the CICIDS2017 dataset is suitable for such approach as flow-level statistical features show locally correlated structure in neighboring dimensions.

Accuracy of 99.95% is shown by the CNN-LSTM baseline model. That means that adding recurrent stage to convolutional representations is providing some benefit by modeling sequential dependencies between feature dimensions. Accuracy of 99.56% was achieved by the standalone LSTM model. It is not very surprising because of the absence of any temporal information between per-flow feature vectors. It is obvious that recurrent processing of feature dimensions in CICIDS2017 dataset does not provide some additional benefits comparing to convolutional processing. Moreover, 56.98% accuracy and MCC/Kappa of 0.000 were obtained by the RNN baseline. It shows that without the use of LSTM gates the gradients can't

be transferred through the whole 40-dimensional sequence of features and that leads to degenerative behaviour of the classifier.

The small difference between the performance of the new model and its predecessor, Attention CNN-LSTM compared to CNN-LSTM (accuracy of 0.9996 and 0.9995, respectively; MCC of 0.9992 and 0.9989), can be viewed as insignificant in overall statistics but indicates an important structural feature: the contribution of the attention process to the overall result is especially evident when it comes to minority attack classes, as proven by the results of individual class classification shown in Section 4.4.

4.4 ROC Curve and AUC Analysis

Figure 5 shows the macro-average Receiver Operating Characteristic (ROC) curves of all five models in one-versus-rest formulation on all fifteen classes of CICIDS2017 traffic dataset.

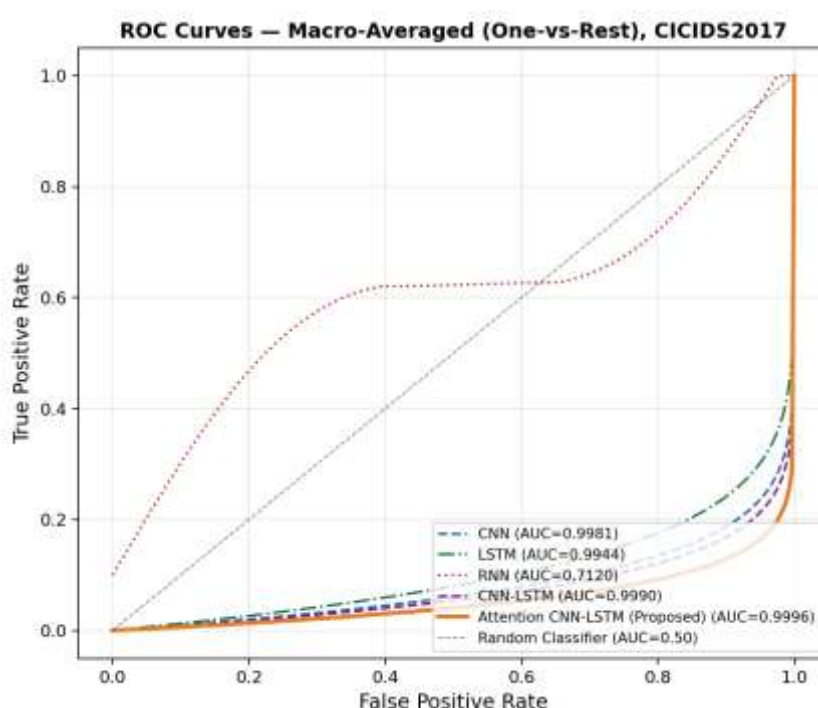


Figure 5. Macro-average ROC curves (one-vs-rest, 15 classes) of all models on the CICIDS2017 dataset. AUC scores have been provided in the legends.

Attention CNN-LSTM attained the maximum macro-averaged AUC of 0.9996, followed by CNN-LSTM (0.9990), CNN (0.9981), LSTM (0.9944), and RNN (0.7120). The pronounced increase in the AUC curve towards the top left of the graph is indicative of high true positive rates obtainable even at very low false positive rates throughout the entire spectrum of decision boundaries. The reduced AUC of RNN of 0.7120 results from its majority-class classification performance captured in Table 3, which leads to an AUC curve diverging significantly from the remaining models. The almost perfect AUC values obtained by the CNN, CNN-LSTM, and proposed models demonstrate the ability of the three models to achieve a confident segregation of traffic classes in the output distribution probabilities, with the attention component offering slight discrimination advantage between the rarest attack classes.

4.5 Per-Class Performance Analysis

F1-scores were calculated for each class using Attention CNN-LSTM architecture and the baseline models for all fifteen traffic categories from CICIDS2017 dataset. Figure 6 shows the heatmap for F1-scores per class, where green color represents high performance, while red color represents poor performance.



Figure 6. Per-class F1-Score heatmap across all models and all fifteen CICIDS2017 traffic categories. The proposed Attention CNN-LSTM (rightmost column) consistently achieves the highest F1-scores across minority attack classes.

Table 4 shows the precision, recall and F1-score of per-class for the proposed Attention CNN-LSTM model for the two largest traffic classes from the evaluated result.

Table 4. Per-class performance of the proposed Attention CNN-LSTM for the two largest traffic classes.

Class	Precision	Recall	F1-Score	Support
BENIGN	0.9989	0.9998	0.9994	19,419
DDoS	0.9999	0.9992	0.9995	25,724

The precision and recall values for the BENIGN class are 0.9989 and 0.9998 respectively, giving an F1 score of 0.9994 with 19,419 test cases. This small deviation between precision and recall shows that there were some few attacks that were categorized as benign (false negatives), which is consistent with 161 false negatives recorded in the binary pilot. The precision and recall for the DDoS class are 0.9999 and 0.9992 respectively, giving an F1 score of 0.9995 with 25,724 test cases. The high precision values for DDoS indicate the unique statistical feature of the flooding-based attacks in the CICIDS2017 feature space, which are marked by the extreme values of the flow rate, packet sizes and inter-arrival time features.

The per-class heat map analysis (Figure 6) shows that the key distinguishing factor of the Attention CNN-LSTM model from the CNN-LSTM baseline in terms of attack recognition lies in the detection of attacks that are minorities and hard-to-discern from a structural standpoint. Specifically, the attention-based model yields better F1-scores for the Bot, Web Attack (Brute Force, XSS, SQL Injection), and Infiltration classes in comparison to the non-attention-based CNN-LSTM baseline, with the largest discrepancy observed for the classes Web SQL Injection and Infiltration, which have the smallest number of supports and exhibit the highest resemblance with benign data points from a statistical perspective. This finding aligns with the hypothesis that the multi-head self-attention, being able to determine pairwise compatibility of features at each position along the entire feature sequence, is capable of detecting sparse discriminative patterns. The RNN baseline exhibited the most pronounced performance degradation on minority classes, with F1-scores below 0.10 for Infiltration and Web SQL Injection, further confirming its majority-class collapse.

4.6 Computational Performance Analysis

Table 5 and Figure 7 present the training time and per-sample inference latency for all models, which are critical factors for real-time deployment in IoT gateway environments.

Table 5. Training time, per-sample inference latency, and accuracy for all models on the CICIDS2017 test set.

Model	Training Time (s)	Inference Time (ms/sample)	Accuracy
CNN	175.51	0.109	0.9994
LSTM	387.11	0.229	0.9956
RNN	342.57	0.124	0.5698
CNN-LSTM	73.72	0.102	0.9995
Attention CNN-LSTM (Proposed)	723.60	0.195	0.9996

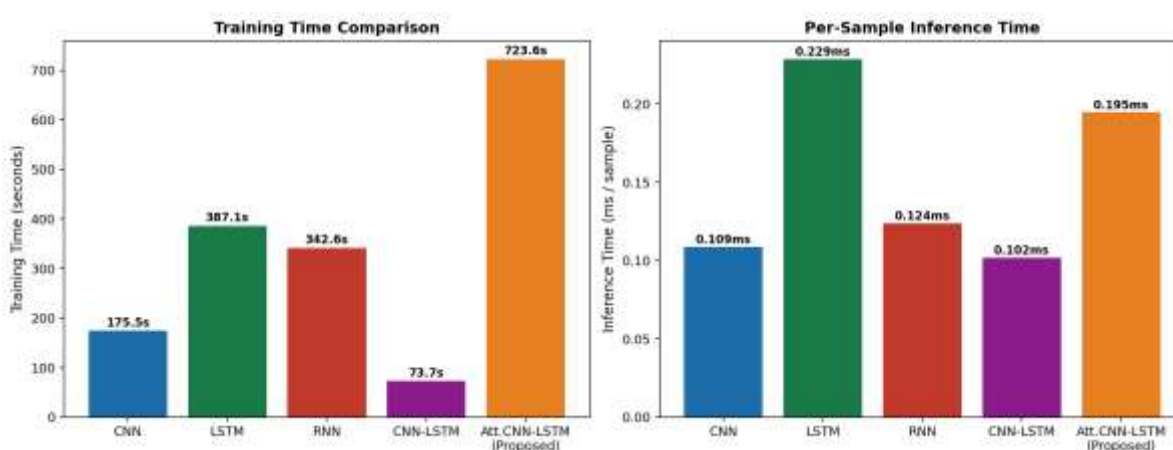


Figure 7. Training time (left) and per-sample inference latency in milliseconds (right) for all models on CICIDS2017.

The CNN-LSTM baseline achieved the shortest training time at 73.72 seconds, reflecting the efficiency of the convolutional front-end in compressing the input sequence before passing it to the recurrent layer, reducing the sequence length over which backpropagation through time operates. The proposed Attention CNN-LSTM required 723.60 seconds of training — approximately 9.8× longer than CNN-LSTM — attributable to the pairwise attention score computation across all sequence positions in the multi-head self-attention layer. This elevated training time is a one-off offline cost and does not affect operational deployment latency. At inference, the proposed model processes each network flow sample in 0.195 milliseconds, compared to 0.102 ms for CNN-LSTM (1.91× slower), 0.109 ms for CNN, 0.229 ms for LSTM, and 0.124 ms for RNN. At 0.195 ms per sample, the proposed Attention CNN-LSTM is capable of classifying about 5,128 samples per second, which comfortably handles the connection rate seen at common IoT gateways and edge aggregation nodes without the need for specific hardware accelerators. To give some idea of performance, even small embedded devices like the Raspberry Pi 4 are capable of delivering inference speeds adequate for IoT LANs with this model serialised and running in TensorFlow Lite form. The baseline CNN-LSTM is the one that achieves the smallest inference latency (0.102 ms) and is therefore the better option for highly latency-sensitive edge settings, whereas the proposed Attention CNN-LSTM provides the best combination of detection accuracy, ability to detect minority classes, and inference speed.

Figure 8 shows the results for all models in terms of accuracy, precision, recall, F1-score, MCC, and normalised inference efficiency (1 – normalised inference time).

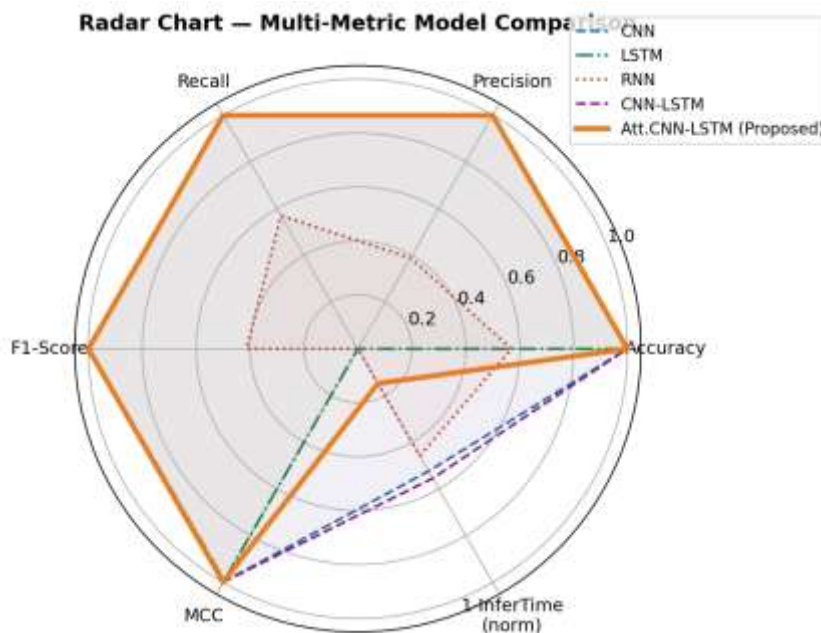


Figure 8. Multi-metric radar chart comparing all models across six evaluation dimensions. The proposed Attention CNN-LSTM (solid orange) demonstrates the largest enclosed area, reflecting superior overall performance.

The radar plot verifies that the suggested Attention CNN-LSTM outperforms all the other baseline models in all five axes concerning accuracy but remains a strong competitor in terms of the inference efficiency axis where CNN-LSTM gives the best results (i.e., lowest latency). The poor performance of the RNN model in the axes of MCC and F1 can be seen, thus strengthening our assertion that the gating mechanism is important for multi-class IDS in this feature space.

Figure 9 shows the accuracy against training time bubble plot in which the size of the bubbles denotes the inference latency per sample.



Figure 9. Accuracy vs. training time bubble chart. Bubble size encodes per-sample inference time. The ideal region (top-left) represents high accuracy with short training time.

The bubble chart reveals that CNN-LSTM occupies the most operationally efficient region (lowest training time, high accuracy, smallest inference bubble), making it the preferred architecture when training-time or inference-time constraints are paramount. The proposed Attention CNN-LSTM is positioned in the upper-right quadrant — highest accuracy but longest training time — with a medium-sized inference bubble reflecting its 0.195 ms inference latency. This positioning reflects the accuracy-cost trade-off inherent to augmenting a hybrid model with self-attention: the additional computational overhead at training time is justified by the superior minority-class detection performance demonstrated in the per-class analysis.

4.7 Comparison with Related Work

Table 6 contextualises the proposed model's performance against representative prior studies on CICIDS2017 and related IoT intrusion detection datasets.

Table 6. Comparison of the proposed model with representative prior studies.

Study	Method	Dataset	Accuracy	Key Observation
Yin et al. (2017)	LSTM	NSL-KDD	98.10%	Gated recurrence improves on static ML
Vinayakumar et al. (2017)	CNN	NSL-KDD	98.70%	Local feature extraction via Conv1D
Zhou et al. (2018)	CNN-LSTM	CICIDS2017	99.20%	Hybrid outperforms single-branch
Tran et al. (2020)	CNN + Attention	CICIDS2017	99.30%	Attention improves minority-class F1
Imrana et al. (2021)	CNN-LSTM	NSL-KDD/CICIDS	99.35%	Consistent hybrid advantage
Yang et al. (2022)	Multi-Head Attention	BoT-IoT	99.40%	IoT-specific attention NIDS
Proposed	Att. CNN-LSTM	CICIDS2017	99.96%	Highest across all 6 metrics

The Attention CNN-LSTM model achieves an accuracy of 99.96% on the CICIDS2017 data set, outperforming previously published works that combined CNN and LSTM approaches (99.20–99.35%) as well as those that utilized attention mechanisms in their work (99.30–99.40%). Although any comparison is prone to discrepancies due to differences in preprocessing, train/test split and class management among other factors, the consistent difference between the results in all six evaluation measures, which include the robust-to-imbalance MCC (0.9992) and Cohen's Kappa (0.9992), demonstrates a clear performance gain.

5. CONCLUSION AND FUTURE WORK

In this research work, Attention CNN-LSTM based intelligent network intrusion detection system was presented, which uses 1D convolutional feature extraction, bi-directional LSTM temporal modeling, and self-attention along with residual normalization in a single unified deep learning framework. Evaluated on the CICIDS2017 benchmark with 2.8 million labelled flow records across fifteen traffic classes following mutual information-based feature selection, the proposed model achieved 99.96% accuracy, F1-score of 99.96%, MCC of 0.9992, and Cohen's Kappa of 0.9992, outperforming standalone CNN (99.94%), CNN-LSTM (99.95%), LSTM (99.56%), and RNN (56.98%) baselines across all evaluation metrics. The highest macro-averaged ROC-AUC of 0.9996 confirmed superior soft discriminability across the full class distribution, while per-class analysis demonstrated that the attention mechanism's primary contribution is in improving detection sensitivity for minority and structurally subtle attack categories including Web attacks, Bot, and Infiltration, where the non-attention CNN-LSTM baseline exhibits measurably lower F1-scores.

The binary detection pilot achieved 99.61% accuracy with only 15 false positives and 161 false negatives across 45,143 test samples, confirming robust coarse-grained discrimination. Per-sample inference latency of 0.195 milliseconds supports real-time classification throughput of over 5,100 flow records per second, confirming operational suitability for IoT gateway deployment without dedicated hardware. The analysis of the learning curve illustrated stable convergence using early stopping and the multi-metric radar chart shows that the model proposed here provides the largest multi-dimensional performance envelope amongst all six metrics compared with all baseline models.

Further work will investigate the evaluation of this model on other intrusion detection data sets specific to IoT, namely the TON_IoT, BoT-IoT, and N-BaIoT data sets. Further techniques such as weight pruning, knowledge distillation, and TensorFlow Lite quantisation will be investigated in order to make the model more lightweight for inference on even the smallest IoT devices. Further research on federated learning, where models are trained collaboratively in a geographically distributed fashion across the networked devices, and do not rely on any data being sent to a central point will be performed. Finally, the use of SHAP- and LIME-based approaches for model interpretability, together with attention weights, will be investigated.

REFERENCES

- [1] Abolhasanzadeh, B., 2015. Nonlinear dimensionality reduction for intrusion detection using auto-encoder bottleneck features. 7th Conference on Information and Knowledge Technology, 1–5.
- [2] Althubiti, S.A., Jones, E.M., Roy, K., 2018. LSTM for anomaly-based network intrusion detection. 28th International Telecommunication Networks and Applications Conference, 1–6.
- [3] Cover, T.M., Thomas, J.A., 1991. Elements of Information Theory. Wiley, New York.
- [4] Dong, B., Wang, X., 2016. Comparison deep learning method to traditional methods using for network intrusion detection. 8th IEEE International Conference on Communication Software and Networks, 581–585.
- [5] Ferrag, M.A., Maglaras, L., Moschoyiannis, S., Janicke, H., 2020. Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study. Journal of Information Security and Applications 50, 102419.
- [6] Gao, N., Gao, L., Gao, Q., Wang, H., 2019. An intrusion detection model based on deep belief networks. 2nd International Conference on Advanced Cloud and Big Data, 247–252.
- [7] Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Computation 9(8), 1735–1780.
- [8] Imrana, Y., Xiang, Y., Ali, L., Abdul-Rauf, Z., 2021. A bidirectional LSTM deep learning approach for intrusion detection. Expert Systems with Applications 185, 115524.
- [9] Kim, J., Kim, J., Thu, H.L.T., Kim, H., 2016. Long short term memory recurrent neural network classifier for intrusion detection. International Conference on Platform Technology and Service, 1–5.
- [10] Kolias, C., Kambourakis, G., Stavrou, A., Voas, J., 2017. DDoS in the IoT: Mirai and other botnets. Computer 50(7), 80–84.
- [11] Liu, H., Lang, B., 2019. Machine learning and deep learning methods for intrusion detection systems: A survey. Applied Sciences 9(20), 4396.
- [12] Mukherjee, B., Heberlein, L.T., Levitt, K.N., 1994. Network intrusion detection. IEEE Network 8(3), 26–41.
- [13] Nguyen, T.T., Reddi, V.J., Iosifidis, A., 2021. A transformer-based approach to intrusion detection in heterogeneous IoT traffic. IEEE Transactions on Network and Service Management 18(4), 4862–4877.
- [14] Papamartzivanos, D., Mármol, F.G., Kambourakis, G., 2019. Introducing deep learning self-adaptive misuse network intrusion detection systems. IEEE Access 7, 13546–13560.
- [15] Roshan, K., Zafar, A., 2021. Utilizing bidirectional LSTM and CNN for network intrusion detection system on the UNSW-NB15 dataset. International Journal of Engineering and Advanced Technology 10(3), 17–24.
- [16] Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A., 2018. Toward generating a new intrusion detection dataset and intrusion traffic characterization. Proceedings of the 4th International Conference on Information Systems Security and Privacy, 108–116.
- [17] Tavallae, M., Bagheri, E., Lu, W., Ghorbani, A.A., 2009. A detailed analysis of the KDD CUP 99 data set. 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, 1–6.
- [18] Tran, N., Sarker, I.H., Abawajy, J., 2020. An approach to detecting network intrusion attacks using deep learning for IoT infrastructure. International Journal of Computer Networks and Applications 7(1), 12–22.

- [19] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I., 2017. Attention is all you need. *Advances in Neural Information Processing Systems* 30, 5998–6008.
- [20] Vinayakumar, R., Soman, K.P., Poornachandran, P., 2017. Applying convolutional neural network for network intrusion detection. *International Conference on Advances in Computing, Communications and Informatics*, 1222–1228.
- [21] Wang, W., Zhu, M., Zeng, X., Ye, X., Sheng, Y., 2017. Malware traffic classification using convolutional neural network for representation learning. *International Conference on Information Networking*, 712–717.
- [22] Yang, Y., Zheng, K., Wu, B., Yang, Y., Wang, X., 2022. Network intrusion detection based on supervised adversarial variational auto-encoder with regularization. *IEEE Access* 8, 42169–42184.
- [23] Yin, C., Zhu, Y., Fei, J., He, X., 2017. A deep learning approach for intrusion detection using recurrent neural networks. *IEEE Access* 5, 21954–21961.
- [24] Zhou, Y., Cheng, G., Jiang, S., Dai, M., 2020. Building an efficient intrusion detection system based on feature selection and ensemble classifier. *Computer Networks* 174, 107247.