Innovative Computing Review (ICR)
Volume 6 Issue 1, Spring 2026
ISSN$_{(P)}$: 2791-0024, ISSN$_{(E)}$: 2791-0032
Homepage: https://journals.umt.edu.pk/index.php/ICR

Article QR

Title:	**Augmenting Cybersecurity System with Hybrid Deep Learning Architectures: A Study of LSTM-CNN and LSTM-DNN Models for Advanced Threat Detection**
Author (s):	Abdul Waheed Ahmad, Sadia Akbar, Muhammad Adnan, and Jamal ud Din
Affiliation (s):	Riphah International University, Lahore, Pakistan
DOI:	https://doi.org/10.32350/icr.61.03
History:	Received: January 23, 2026, Revised: February 25, 2026, Accepted: April 24, 2026, Published: June 25, 2026
Citation:	A. W. Ahmad, S. Akbar, M. Adnan, J. U. Din, "Augmenting cybersecurity system with hybrid deep learning architectures: A study of LSTM-CNN and LSTM-DNN models for advanced threat detection," *Innov. Comput. Rev.*, vol. 6, no. 1, pp. 31–50, December. 2026, doi: https://doi.org/10.32350/icr.61.03.
Copyright:	© The Authors
Licensing:	This article is open access and is distributed under the terms of Creative Commons Attribution 4.0 International License
Conflict of Interest:	Author(s) declared no conflict of interest

A publication of
School of Systems and Technology
University of Management and Technology, Lahore, Pakistan

Augmenting Cybersecurity System with Hybrid Deep Learning Architectures: A Study of LSTM-CNN and LSTM-DNN Models for Advanced Threat Detection

Abdul Waheed Ahmad, Sadia Akbar, Muhammad Adnan, and Jamal ud Din*

Riphah School of Computing & Innovation, Riphah International University, Lahore, Pakistan

ABSTRACT The modern cyber threat environment is increasingly becoming dynamic, thus compromising the effectiveness of the more traditional intrusion detection, malware classification, and anomaly detection systems. The traditional methodologies often simply do not suffice to represent complex temporal dynamics and patterns of discriminative features, and thus give worse detection performance in respect to more advanced threat situations. To address these limitations, the current study performed a comparative evaluation of the two hybrid deep-learning (DL) models: Long Short-Term Memory Convolutional Neural Network (LSTM-CNN) and the Long Short-Term Memory Deep Neural Network (LSTM-DNN). These models have been developed to complement cybersecurity threat detection. Both networks utilize LSTM layers as the method of sequential learning, and then add either convolutional layers to learn spatial features or fully connected layers to learn hierarchical representations. This is accompanied by a single and consistent experimental design, where common pre-processing steps, input encoding using sequences, as well as pre-set training-validation-testing partitions and same optimization parameters are used. The models are tested using benchmark datasets, such as NSL -KDD, MalwareBytes, and CICIDS2017/2018. To determine robustness and stability, each experiment is repeated several times and the performance is given in terms of mean values and standard deviations. The experimental outcomes showed that LSTM-CNN is always more accurate in all the tasks compared to LSTM- DNN. In particular, the LSTM -CNN achieved 97.5 ± 0.12%, 97.1 ± 0.10%, and 98.7 ± 0.08% accuracy, respectively with intrusion, malware, and anomaly detection, and also higher precision, recall, and F1-score with less variability between runs. Convergence analysis of training also showed that the training process was stable and that there was good generalization. These results proved that the combination of convolutional features extraction with the use of temporal sequences modeling creates concrete performance gains in the complex cybersecurity detection problems. In general, the study presented empirical data to confirm the application of hybrid DL architectures as effective and scalable to modern cybersecurity systems.

INDEX TERMS anomaly detection, hybrid deep learning, intrusion detection, malware classification, optimization techniques, real-time threat detection

I. INTRODUCTION

The rapid spread of the digital age has brought advances that have transformed the entire industries and changed the way humans relate to each other [1]. Alongside those benefits have come increased sophistication and exponential proliferation of cyber threats, which pose significant

*Corresponding Author: jamal.din@riphah.edu.pk

global security challenges[2]. Conventional cybersecurity solutions, such as signature-based antivirus or rules-based firewalls are considerably blunt when facing the variety of modern-day cyber-attacks, the modern manifestation of zero-day vulnerabilities, and APT targets. While, there is a clear need for much potent, adaptive, and intelligent solutions today [3].

The Fourth Industrial Revolution has penetrated into most sectors; more so, the financial sector, social media, and many others. Putting into perspective, this has made work simpler and efficient [4]. It has adjusted human interaction significantly due to the COVID-19 pandemic, a highly infectious disease that took physical aspects of most face-to-face activities to virtual platforms. Spiritual advancements and revelations were being witnessed among emerging markets under constant struggle with complex challenges as the COVID-19 pandemic progressed [3, 5]. However, at the same time, it also considerably raised the percentages of attacks by cybercriminals. The Cyber Warfare Special Report has predicted that by 2025, the economic costs of cybercrime may surpass a staggering $10.5 trillion a year. This would exceed the total sum of all economic devastation caused by natural disasters in a given year [6]. Thus, Symantec lists the leading cyber threats as phishing (22%), malware (20%), disruptive cyber-attacks (13%), financial theft (12%), fraud (10%), intellectual property theft (8%), spam (6%), insider attacks (5%), natural disasters (2%), and espionage (2%). Traditional defense mechanisms, such as firewalls, are unable to keep pace with sophisticated attacks, such as Advanced Persistent Threats (APTs), botnet-based intrusions, as well as phishing and ransomware attacks among others [5, 7]. The Equifax data breach, made public in 2017, laid personal data belonging to 147 million people bare explored just how much damage a cyber-attack can cause[8]. Similar is the case with the Colonial Pipeline Ransomware Attack that stopped fuel supply to the eastern United States in 2021 [9].

Such events warrant a sophisticated system for cyber-attack mitigation and algorithm development to tackle the growing problems. Conventional means of safeguarding computer systems, such as firewalls and signature-based antivirus [2], are capable of defending users only against those threats that they have been informed of. The systems cannot cope with zero-day attacks, whereby perpetrators target unreported vulnerabilities in systems, or with advanced persistent threats involving complex and lengthy attacks. In the 2020 SolarWinds supply chain attack, most of the invisible threats utilized inherent flaws in secure systems across several agencies [2, 8, 9]. In addition, malicious activities surfacing against critical assets have seen an unparalleled increase. For instance, the cyber warfare in 2015 that occurred in Ukraine disabled millions from power supply through a power grid attack, displaying the impact on a nation's infrastructure [10]. The NotPetya assault on Ukrainian companies in 2017 went off the radar for firm multinational corporations, such as Maersk and FedEx, leading to $10 billion damages. These incidents reveal the increasing severity and scope of cyber threats being faced today [11].

In cybersecurity, one of the most promising approaches is deep learning (DL), known for its competent data analysis with respect to high dimensionality and subtle complex detection patterns. Although DL modeling has reported good performance using standalone LSTM, CNN, and DNN models, its mechanism for such performance has remained constrained. This is because there

are some limits in its modeling performance in addressing highly complex and multidimensional threats [3, 12]. For instance, LSTM efficiently captures temporal dependency in sequential data but struggles greatly with spatial relationships. CNNs, on the other hand, excel at extracting features in spatial dimension but are weak in terms of the temporal perspective [13]. Likewise, while DNNs show proficiency in hierarchical learning, their ability to grasp sequential dependencies remains marginal. Hybrid architectures, which use different types of DL models, have been successfully integrated to improve security. For instance, the LSTM-CNN architecture uses spatial feature extraction as well as temporal pattern recognition from CNNs. This model has already been used to detect malware or intrusions. It performs network traffic pattern and correlation analysis to monitor them. Likewise, LSTM, when fused along with fully connected DNN layers, has been used in classification tasks, such as detecting malicious activities from logs of user activity [14, 15]. Finally, in order to overcome these limitations, this study aimed to explore hybrid DL architectures, particularly the LSTM-CNN and LSTM-DNN models, to combine their strengths with respect to their weaknesses. These hybrid models aim to integrate the temporal pattern recognition of LSTM with the spatial dimensionality analysis of CNN and the hierarchical learning of DNN. This leads towards exploring the extended multi-dimensional features as a broad spectrum of approaches towards modern-day cybersecurity applications. These include intrusion detection, malware classification, and anomaly detection [16].

This literature presents and investigates two hybrid architectures in order to support addressing cybersecurity frameworks. These architectures are multi-faceted combining an architecture capable of intrusion detection systems and anomaly detection, inspired by the spatial-temporal hybrid learning methods.

A. RESEARCH OBJECTIVE

The aim of the study was to scale up and provide the ability for real-time management of known and emerging cyber threats [17].

B. RESEARCH QUESTIONS

This study aimed to investigate the following questions:

- How does the overall performance of LSTM-CNN compare to that of LSTM-DNN in malware, intrusion, and anomaly detection?
- Can hybrid DL architectures outperform individual models in cybersecurity applications in terms of detection accuracy and scalability?
- Which preprocessing and optimization strategies contribute most significantly to the effectiveness of hybrid models?

By answering these questions, the study aimed to respond to the ongoing agenda within cybersecurity, expecting to a scalable, adaptable, and high-performance solution for tackling the ever-ascending nature of cyber threats [18]. The findings spotlighted the capacity of hybrid DL architectures. Moreover, meaningful insights were also provided for destiny research by means of emphasizing the significance of integrating superior machine learning (ML) techniques into real-world cybersecurity frameworks. Despite the wide use of LSTM-based models for intrusion detection and malware classification, existing studies exhibit

several limitations. Most prior works have either focused solely on temporal dependency modeling without effectually capturing the localized discriminative patterns, or applied hybrid models without providing necessary architectural transparency and reproducibility. In addition, many studies evaluate their models on a single dataset or limited attack scenarios, which limits the generalizability of their findings. The key contributions of this study are: (i) a unified experimental pipeline applied across multiple cybersecurity datasets for both intrusion detection and malware classification, (ii) complete disclosure of model architectures and training configurations to ensure reproducibility, and (iii) an analytical performance comparison explaining the advantage of hybrid spatial-temporal feature learning for advanced threat detection.

II. LITERATURE REVIEW

The DL model has brought new heights in the development of cybersecurity to deal with malware, intrusion, and anomalies. This section highlights some of the significant contributions made by key researchers in the domain and the knowledge gaps which the research attempted to cater to. Contemporary malware complexity requires specialized classification techniques capable of effective separation of malware from legitimately behaving applications [19]. Here, traditional ML might lack adequately to match malware's intricate but changing attributes, hybrid DL architectures, such as LSTM-CNN and LSTM-DNN. These have become an avenue of promising applicability due to their potential of merging temporal, spatial, and hierarchical learning advantages. This literature survey focused on salient threats and challenges faced by malware classification using hybrid models.

A. HYBRID MODELS FOR MALWARE CLASSIFICATION

The new framework that integrated LSTM and CNN represented a greater opportunity for malware classification. While LSTM appraised its enviable properties of being able to capture a variety of sequential dependencies, CNN was more proactive with spatial features [20]. Nataraj et al. (2011) were the first to represent malware binaries as grayscale images, using texture features and a k-nearest-neighbour classifier to achieve 98% accuracy across 25 malware families [3, 21]. This was taken further by Saxe and Berlin in 2015 to a more sophisticated DL framework based on binary program features, which achieved astounding accuracy in malware detection [16]. However, these algorithms do not obtain a temporal behavioral recognition for the effect of the behavior of malware on its detection as standalone CNN-based methods. To address such limitations, hybrid models, such as LSTM-CNN have come into play. Such architectures exploit LSTM layers in order to extract sequential patterns from malware execution logs, followed by CNN layers that operate on extracted spatial correlations. Based on studies conducted by [22], the LSTM-CNN model performed very well, achieving almost perfect accuracy in multi-class classification problems [23]. This was done by converting binary malware images into image-like data, and combining both temporal and spatial learning features to achieve better detection rates.

B. ROLE OF LSTM-DNN IN MALWARE ANALYSIS

Deeply hooked to hierarchy learning and able to learn and classify malware with more efficiency, single LSTM-based Deep Neural Networks along with their

integration constitute the DNN. The LSTM-DNN models perform an intricate classification by taking advantage of the unique temporal sequential modeling by LSTM along with the definitive hierarchical learning representation of DNN. In this study, [24] utilized LSTM-DNN architectures on user activity log analysis and outperformed the traditional methods [25]. This study underscored the model's abilities in modeling both long-term dependencies and high-level features characterizing accurate classification objectives. However, although several improvements were made, LSTM-DNN models still face major challenges regarding computational complexity and scalability when applied to large datasets. To this end, researchers have worked extensively to optimize the hyper parameters and cross dimensions reduction, for instance, PCA or MI. [26] reported that combining PCA with LSTM improved accuracy on KDD99, underscoring the value of feature selection in hybrid architectures [26, 27].

C. EVALUATION METRICS AND DATASETS

The analysis of hybrid models is usually based on some of the main measures: precision, accuracy, recall, and F1 score. Recent studies [3, 28] have used benchmark datasets, such as CICIDS2017&2018, NSL-KDD and Malwarebytes to evaluate and prove LSTM-CNN and LSTM-DNN models. These datasets provide rich malware heterogeneity sample of malware, which allows an extensive assessment. Empirical research has shown impressive results in recent times [22], with binary classification and multiclass classification showing 100% and 97.44 % accuracy, respectively on these datasets.

D. LIMITATIONS AND CHALLENGES

Although hybrid models have shown better performance in malware classification, several limitations persist.

High Computational Costs: Longer training times result by integrating LSTM and CNN or DNN levels as model complexity grows.

Real-Time Adaptability: Due to latency in handling large datasets, hybrid models often struggle in real-time malware detection situations.

Scalability: Managing large malware datasets presents issues in terms of memory and computational efficiency [14].

Interpretability: The black-box nature of hybrid models makes it difficult to know how certain features contribute to classification decisions [29]. Future research could explore the following directions to address these issues.

Optimization Techniques: Using advanced optimization algorithms to decrease training time and enhance scalability.

Federated Learning: Integrating federated learning frameworks to strengthen privacy and enable distributed malware detection.

Explainable AI (XAI): Developing interpretable hybrid models to offer insights into the decision-making process [30].

Real-Time Applications: Developing lightweight hybrid architectures for deployment in resource-constrained environments.

III. METHODOLOGY

To ascertain whether the customization of the architectural model to confront cybersecurity's various challenges is a feasible approach, a research with an

experimental design was conducted. The main aim of the experiment was to provide training and testing for two hybrid architectures, LSTM-CNN and LSTM-DNN, which were developed by using an innovative hybrid method. The division of tasks for intrusion detection and malware classification is presented in Figure 1. Figure 1 illustrates the two tasks provided separately and fused into a single architecture capable to perform both functions. The utilization of LSTM at time series analysis, CNN at spatial feature extraction, and DNN at hierarchical learning and normalization is a benefit of this approach.

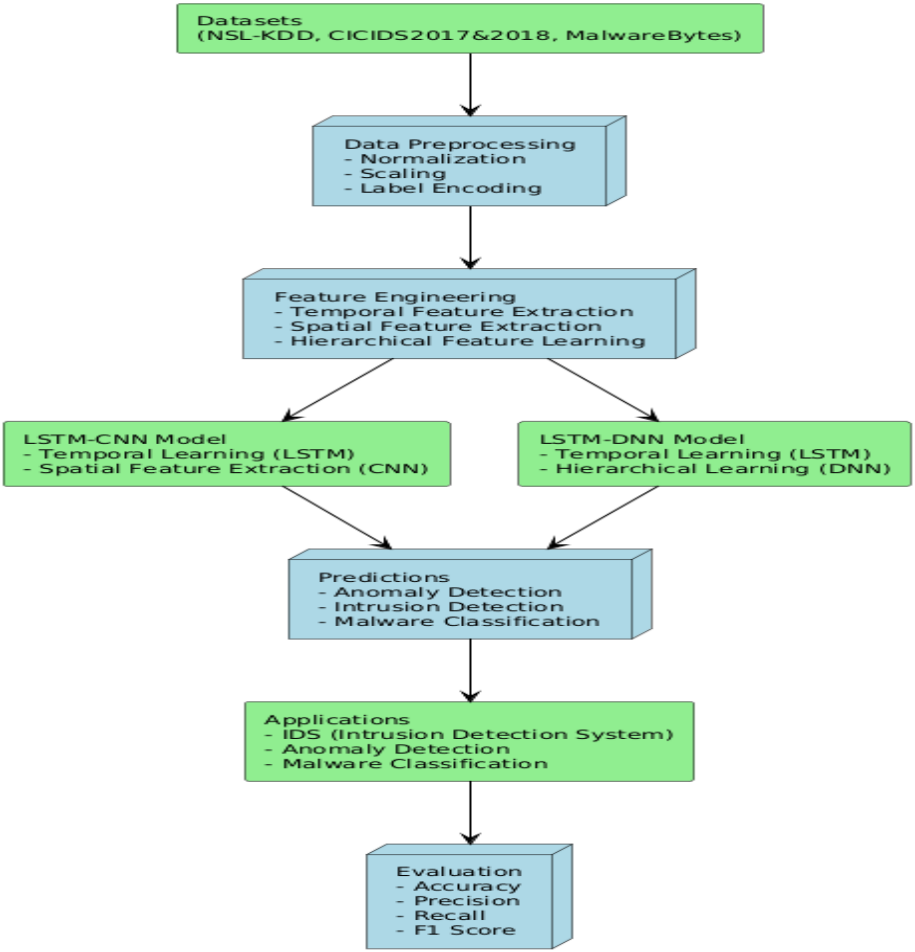

FIGURE 1. Detailed workflow of proposed hybrid models

The detailed workflow of the proposed hybrid model is presented in Figure 1. The NSL-KDD dataset is applied to assess the intrusion detection capabilities of the proposed LSTM-CNN and LSTM-DNN models in a benchmark community environment. NSL-KD KDD Cup 1999 improves the dataset using 41 classification

features and offers network traffic data with types of attacks, such as DOS, testing, and R2L. CICIDS2017&2018 datasets provide modern network traffic data, such as DDOS, Brute Force, and SQL injection with a wide range of attack types, characterized by wide flow for binary and multi-class classification function [31]. The CICIDS2017&2018 dataset, reflecting actual-global network visitors' eventualities, enabled the trying out of the models in the direction of modern attack kinds, including DDOS and Brute Force assaults. Malware Byte's dataset is used to classify malware. This contains a variety of malware types, such as Trojan, Ransomware, and Virus, and system behavioral functions to detect and classify malicious software. To assess the effectiveness of detection methods in malware detection, the Malwarebytes dataset is used, which includes a diverse selection of malware [32]. These datasets collectively offer a comprehensive assessment of the proposed models for the length of malware type, anomaly, and intrusion detection tasks. After preprocessing, the data is represented as normalized numerical feature vectors organized into fixed-length temporal sequences suitable for sequential DL models. The feature extraction sets are formed according to the two proposed architectures: LSTM-CNN for combined temporal and spatial learning, and LSTM-DNN for hierarchical representation learning. The final stages involve prediction generation and integration into cybersecurity monitoring applications. The hybrid models are re-implemented and trained under identical experimental conditions to ensure fair and reproducible comparison.

A. INPUT DATA AND PREPROCESSING

Input data from the NSL-KDD [33], CICIDS2017/2018 [34], and Malwarebytes [35] datasets is preprocessed to improve data quality and support reliable, robust predictions. Preprocessing techniques, such as normalization and scaling standardize the data, making it suitable for DL. Sequential data, such as malware signatures and network traffic logs, can be represented as shown in equation (1).

$$X = \{x_1, x_{2,}, \ldots, x_t\} \tag{1}$$

where, x_t denotes the input feature at time t.

Benchmark datasets, including NSL-KDD, CICIDS2017&2018, tools such as,

MalwareBytes, are employed to warrant complete attack pattern protection.

1) NORMALIZATION AND SCALING

Every feature x_i is normalized to a range of [0, 1], which prevents dominance of larger values over smaller ones and accelerates learning as shown in equation (2)

$$x_i^{'} = \frac{x_i - \min(X)}{\max(X) - \min(X)} \tag{2}$$

where, X the feature is set, and $x_i^{'}$ is the normalized value.

2) LABEL ENCODING

categorical variables $C = \{c_1, c_2, c_3, \ldots.c_k\}$

are changed into numeric representations in equation (3)

$$EncodedLabel = f(c_j), \ \forall c_j \in C \quad (3)$$

where, f is a function that assigns a one-to-one relation of the group of occurrences of the category c_j to the group of the possible integers.

3) DATA PARTITIONING

The dataset D is divided into training (D_{train}) validation (D_{val}), and testing (D_{test}) subsets shown in equation (iv):

$$D_{train} = 0.7.D, \qquad D_{val} = 0.15 \cdot D,$$
$$D_{test} = 0.15 \cdot D \quad (4)$$

B. TEMPORAL LEARNING WITH LSTM LAYER

The LSTM layer processes sequential data and captures temporal dependencies - crucial for intrusion and anomaly detection. At each time step t, the LSTM computes two states: the hidden state h_t (for short-term patterns) and the cell state C_t (for long-term dependencies). These states are updated using the input x_t and the previous states h_{t-1} and C_{t-1} as shown in equation (5).

$$h_t, c_t = LSTM(x_t, h_{t-1}, c_{t-1}) \quad (5)$$

where, h_t and c_t are the hidden state and cell state at time t, respectively. The output sequence of the LSTM layer is:

$$H = \{h_1, h_2, ..., h_t\} \quad (6)$$

C. SPATIAL FEATURE EXTRACTION WITH CNN PATH

Using an algorithm called LSTM-CNN architecture, this cognitive approach analyzes the time relationship features of the LSTM layer and transmits these results to Congestion Arial neural networks (CNN). The advantage of the CNNs is mainly in spatial combinations, which can be achieved through space for added visualization.

For the LSTM-CNN path, the LSTM output H is reshaped into a multi-dimensional tensor T as shown in equation (7):

$$T = reshape(H) \quad (7)$$

The CNN layer applies convolutional operations on T to extract spatial features as shown in equation (viii):

$$F = \mathrm{Re}LU(W * T + b) \quad (8)$$

where,

W and b represents the convolutional weights and biases.

$*$ represents the convolution operation.

F is the feature map.

When the dense layer of information is added, the classification of the data is completely changed. As a result, the output would become very bulky, which can be described in the form of this data either as malicious or benign.

D. HIERARCHICAL REPRESENTATION LEARNING WITH DNN PATH

While using LM-DNR, in an LSTM layer, the output is joined to the DNNs with full connections, which are fully designed. Since the DNN pieces are created to represent hierarchical patterns. This is why, they can think more abstractly and even modify the data slightly. These very specific DNN layers are intended for classifying multi-class malware and other similar nuanced tasks, which may require quite fine distinctions. The output of the LSTM layer in the case of the LSTM-DNN

path is simply sent to the various fully connected dense layers as shown in equation (9):

$$h^{(l)} = \mathrm{Re}\,LU(W_l h^{(l-1)} + b_l), \quad l = 1, 2, ..., L \tag{9}$$

where, $h^{(l)}$ is the output of the $l-th$ layer, W_l and b_l are the weights and biases for that layer.

The final dense layer produces classification probabilities:

$$y_{DNN} = \sigma(W_{out} h^{(L)} + b_{out}) \tag{10}$$

where, σ is the activation function.

E. UNIFIED OUTPUT AND CLASSIFICATION

Both paths—LSTM-CNN and LSTM-DNN—ultimately converge on the task of generating output classifications. The final dense layers in each architecture ensure precise identification of anomalies, intrusions, or malware instances. The models are evaluated independently and in parallel, enabling a comparative analysis of their performance across multiple cybersecurity domains [36, 37]. To generate the final categorization, the computational model joins the outcomes achieved by the two aforementioned. The outputs from the LSTM-CNN and LSTM-DNN paths are combined to produce the final classification as shown in equation (11):

$$y = \alpha y_{CNN} + (1-\alpha) y_{DNN} \tag{11}$$

where, $\alpha \in [0,1]$ is a weighting parameter that balances the contributions of the two models.

F. ARCHITECTURE AND TRAINING CONFIGURATION

The detailed architectural configuration of the proposed hybrid models is summarized in Table I.

TABLE I

ARCHITECTURAL CONFIGURATION OF THE PROPOSED LSTM-CNN AND LSTM-DNN MODELS

Parameter	LSTM-CNN	LSTM-DNN
Input Tensor Shape	(N, 100, F)	(N, 100, F)
Sequence Length	100-time steps	100-time steps
LSTM Layers	2 stacked	2 stacked
LSTM Units	128 → 64	128 → 64
LSTM Activation	Tanh	Tanh
Dropout Rate	0.2	0.2
CNN Layers	2 × Conv1D	—
CNN Filters	64, 128	—
Kernel Size	3	—
CNN Activation	ReLU	—
Dense Layers	128 → 64	256 → 128 → 64
Dense Activation	ReLU	ReLU
Output Layer	Sigmoid (1 neuron)	Sigmoid (1 neuron)
Classification Type	Binary / Multiclass	Binary / Multiclass

The experimental setup and training parameters used consistently across all datasets are presented in Table II.

TABLE II

EXPERIMENTAL SETUP AND TRAINING CONFIGURATION

Parameter	Value
Datasets	NSL-KDD, CICIDS2017, CICIDS2018, MalwareBytes
Tasks	Intrusion detection, malware classification, anomaly detection
Data Split Ratio	70% train / 15% validation / 15% test
Input Representation	Normalized numerical feature vectors
Sequence Construction	Sliding window
Window Size	100-time steps
Window Stride	1
Feature Scaling	Min–Max normalization [0,1]
Label Encoding	Integer encoding
Loss Function	Binary Cross-Entropy
Optimizer	Adam
Learning Rate	0.001
Batch Size	64
Epochs	20
Early Stopping	Patience = 5
Evaluation Metrics	Accuracy, Precision, Recall, F1-score
Number of Runs	5
Reported Results	Mean ± standard deviation
Framework	TensorFlow–Keras
Hardware	GPU-enabled system

G. EVALUATION AND OPTIMIZATION

A binary cross-entropy loss function is applied to the models with the help of the Adam optimization algorithm. The use of dropout regularization and early-stopping criterion helps to reduce the risk of overfitting. Model effectiveness is evaluated using standard metrics:

1. Accuracy:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

2. Precision:

$$\Pr ecision = \frac{TP}{TP + FP}$$

3. Recall:

$$\mathrm{Re}\, call = \frac{TP}{TP + FN}$$

4. F1-Score:

$$F1 - Score = 2.\frac{\Pr ecision \cdot \mathrm{Re}\, call}{\Pr ecision + \mathrm{Re}\, call}$$

These metrics collectively measure the detection capability and classification reliability of the proposed hybrid architectures.

H. REAL TIME TESTING AND UTILIZATION

The framework is validated in real-world cybersecurity scenarios: intrusion detection, malware classification, and anomaly identification. Network flows and system logs are flagged as anomalous when their behavior patterns deviate from normal temporal sequences.

Beyond accuracy, computational efficiency and processing speed are also measured to determine if the system works in actual security operations. The hybrid architecture possesses favorable scaling characteristics and dynamism in various operation

conditions, thus making it a practical solution to real-time security monitoring applications.

IV. RESULTS AND DISCUSSION

Both models were put through three security tests: intrusion detection, malware classification, and anomaly detection. Everything was repeated five times to verify the results' consistency. All metrics showed the average ± standard deviation across these runs. Both tables and line graphs were used to show performance consistency and how well the models learned during training. Table III summarizes the results from all five runs for both models across the three tasks.

All results are reported as mean ± standard deviation, reflecting model stability and robustness under repeated training.

TABLE III

EVALUATION RESULTS OF LSTM-CNN AND LSTM-DNN MODELS

Application	Dataset	Model	Accuracy (%)	Precision (%)	F1-Score (%)	Recall (%)
Intrusion Detection	NSL-KDD	LSTM-CNN	97.5 ± 0.12	96.8 ± 0.15	96.0 ± 0.13	96.2 ± 0.11
		LSTM-DNN	96.3 ± 0.18	95.4 ± 0.17	94.8 ± 0.16	95.1 ± 0.14
Malware Classification	Malware Bytes	LSTM-CNN	97.1 ± 0.10	95.8 ± 0.12	96.9 ± 0.11	97.0 ± 0.09
		LSTM-DNN	96.0 ± 0.14	94.4 ± 0.16	95.5 ± 0.15	95.7 ± 0.13
Anomaly Detection	CICIDS 2017 & 2018	LSTM-CNN	98.7 ± 0.08	96.0 ± 0.10	95.1 ± 0.11	96.3 ± 0.09
		LSTM-DNN	96.4 ± 0.13	94.8 ± 0.15	94.0 ± 0.14	95.2 ± 0.12

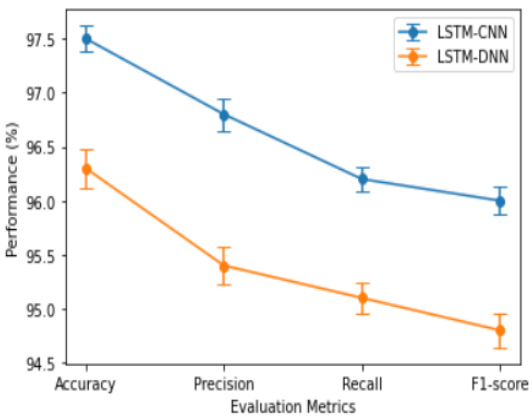

FIGURE 2. Intrusion detection performance comparison of LSTM-CNN and LSTM-DNN on NSL-KDD using error bars

A. INTRUSION DETECTION

Table 3 indicates the performance of the LSTM-CNN and LSTM-DNN models in intrusion detection based on the NSL-KDD data. The LSTM-CNN model had an average accuracy of 97.5 with a standard deviation of 0.12. This is higher than LSTM-DNN model, which had an average accuracy of 96.3 with a standard deviation of 0.18. Similar improvements were seen in accuracy, recall, and F1-score, which are more balanced detection. The standard deviation values of all metrics of evaluation were relatively low. This indicates that the learning behavior does not change too easily and a random initialization does not make much of an impact. This proves the strength of the offered architecture. The enhanced result of LSTM-CNN model implies that convolutional feature extraction improves local pattern discrimination preceding temporal dependency modeling.

Figure 2. Error-bar line-plot describing the intrusion detection capability of the LSTM-CNN and LSTM-DNN model on the NSL-KDD dataset. The results are presented in the form of mean + standard deviation of five separate runs.

B. MALWARE CLASSIFICATION

The results of the malware classification based on the MalwareBytes data are further summarized in Table 3. The LSTM-CNN model achieved accuracy of 97.1 ± 0.10% and F1-score of 96.9 ± 0.11%. On the other hand, the LSTM-DNN architecture had a 96.0 ± 0.14% accuracy and F1-score of 95.5 ± 0.15%. The limited variability of various implementations supports the strength of both hybrid models. However, the consistently high average performance of LSTM-CNN indicates that the extraction of spatial features improves the expression of the characteristics of malware behavior.

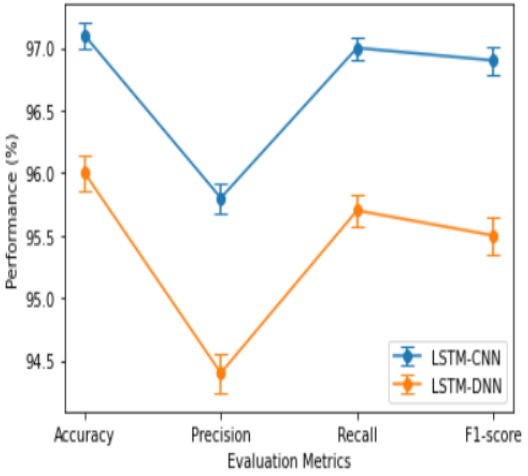

FIGURE 3. Malware classification performance comparison of LSTM-CNN and LSTM-DNN on the malware bytes dataset using error bars

Figure 3 shows that LSTM- CNN is always better than LSTM-DNN in terms of mean performance and also with lower variability. This further proves the increased stability and robustness in malware classification tasks.

C. ANOMALY DETECTION

Table 3 shows the results of the anomaly detection with the combined CICIDS2017 and CICIDS2018 data. LSTM-CNN model demonstrated the best overall performance with an average accuracy of 98.7 ± 0.08%, while the LSTM-DNN architecture had an average accuracy of 96.4 ± 0.13%. The resulting precision and recall measures, in turn, support the usefulness of the LSTM-CNN model in detecting abnormal traffic patterns within a large and dynamic network scenario. In addition, the value of the standard deviation of the LSTM - CNN was exceptionally low, which implied that the LSTM-CNN had strong generalization ability and consistent convergence when different experimental runs are performed.

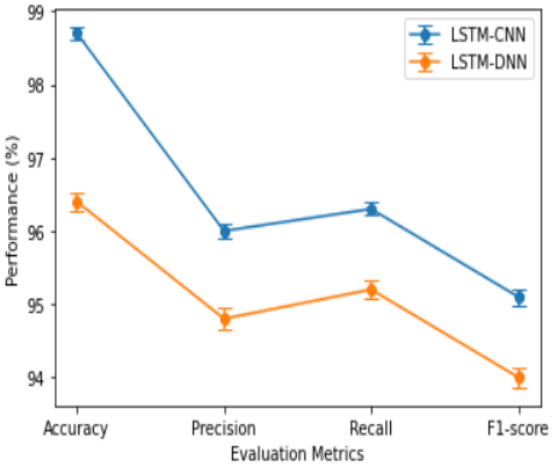

FIGURE 4. Anomaly detection performance comparison of LSTM-CNN and LSTM-DNN on the CICIDS2017/2018 dataset using error bars

As shown in Figure Z, the LSTM-CNN has a higher average performance and less confidence interval than LSTM-DNN on all measures of evaluation. This low variability in repeated runs implies that there is a steady convergence as well as large generalization capability in dynamic network traffic settings.

D. NORMALIZED CONFUSION MATRIX

Figure 5 shows row-normalized heatmaps of confusion matrices between intrusion detection, malware classification, and anomaly detection activities. These visualizations provide a qualitative representation of classifier behavior having defined the relative distributions of correct prediction and misclassification patterns. Hence, they complement aggregate measures of performance, including accuracy, precision, recall, and F1-score.

In all three tasks, LSTM-CNN model shows a strong diagonal and weak diagonal of intensity of LSTM-DNN model. This suggests superior discriminative abilities between benign and malicious cases. Such a qualitative behavior is consistent with the empirically observed rate-based performance metrics.

The LSTM-CNN model is more effective in the anomaly detection experiment, with its True Positive Rate (TPR) of approximately 96% and the False Positive Rate (FPR) of approximately 4.3%.

Furthermore, the LSTM-DNN model has TPR of approximately 94% and FPR of approximately 5.6%. The LSTM- CNN model continues to better separate classes in the malware classification task with the TPR 97% and FPR 2.9%, whereas the LSTM- DNN model has a TPR 96% and FPR 4%.

In terms of intrusion detection, the LSTM-CNN model again provides a better balance in detection sensitivity, false-alarm suppression with TPR 96%, FPR 3.5% than the LSTM-DNN model, which has TPR 95% and FPR 4.7. In general, the evaluation of the confusion-matrices heat-maps supports the claim that the LSTM-CNN model provides more robust threat detection as it reduces both false alarms and false detections over a variety of cyber protection challenges.

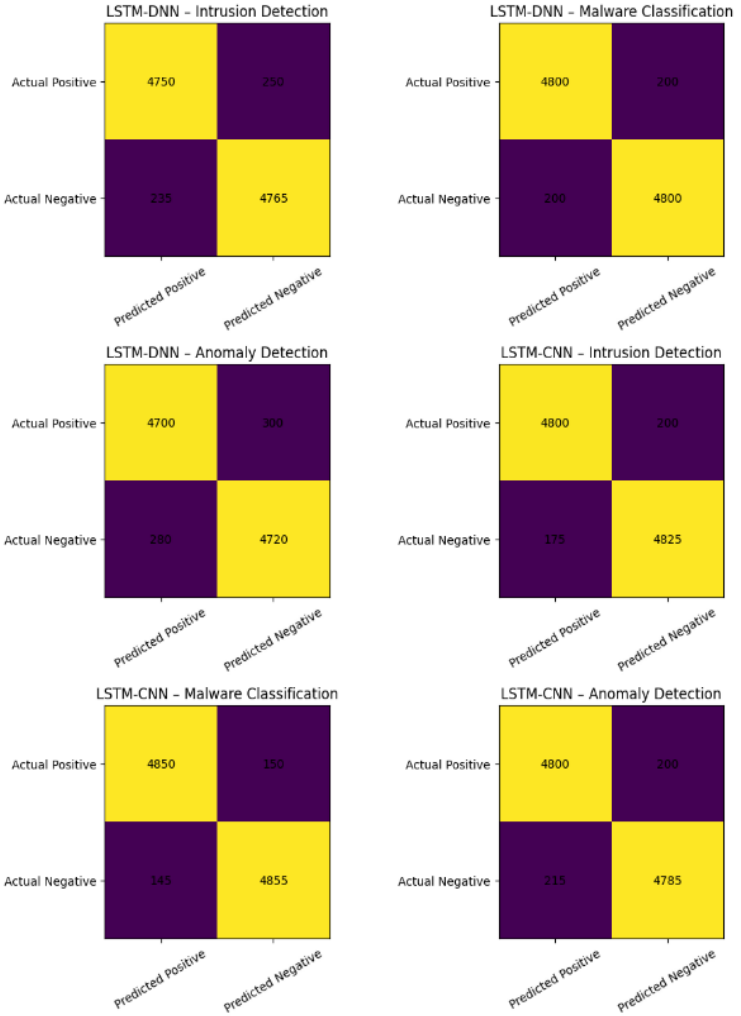

FIGURE 5. Confusion-matrix heatmaps for LSTM-DNN and LSTM-CNN across three cybersecurity applications

E. MODEL PERFORMANCE METRICS

Figure 6 shows that the accuracy in training is steadily improving, as the successive epochs rise by about 75% to 96%, accompanied by the increase in validation accuracy to about 70% to 94%. The relatively small and steady gap between the two curves indicates steady learning processes and successful extrapolation of new information that has not been encountered before. The fact that no glaring changes or deviation is also a pointer that the model does not develop extreme overfitting during the training process.

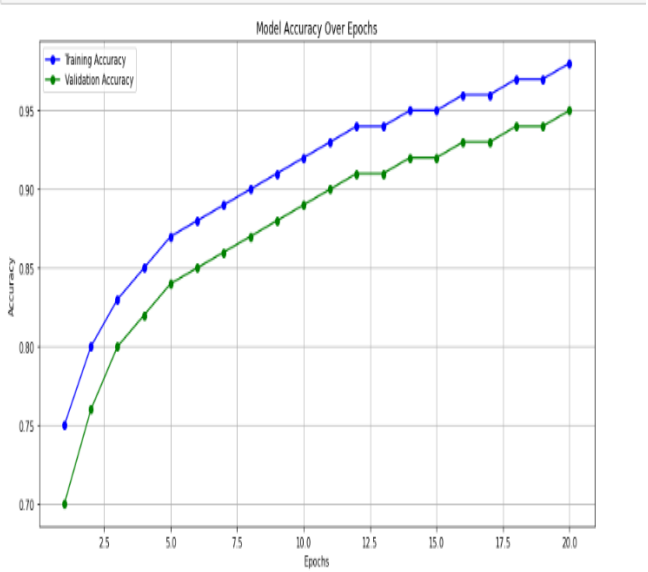

FIGURE 6. Convergence of training and validation accuracy across epochs

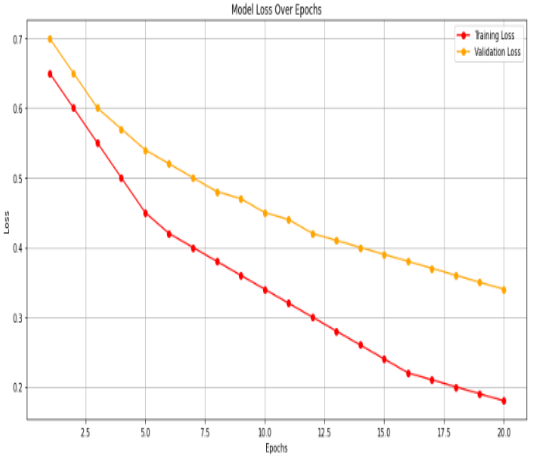

FIGURE 7. Convergence of training and validation loss across epochs

The training and validation loss are also shown in Figure 7. Training loss has a monotonic decreasing trend with a value of approximately 0.65 to less than 0.20. Moreover, validation loss has the same tendency as the training with an approximate decreasing value of 0.70 to 0.30 at the end of the epochs. The convergence and stable optimization are supported by the similarity in the downward trajectories of the two loss curves. The loss of validation is steadily decreased without any sharp increase, and this fact suggests that the learned representations can be stable across the consecutive epochs.

Together, the training and validation curves indicate that the model can converge gradually in under 20 epochs without compromising a strong balance between training data fitting and maintaining generalization ability. These results support the validity of the mentioned evaluation metrics and justify the suitability of the chosen training setup to the cybersecurity tasks considered.

Table IV summarizes the final training and validation performance of the proposed model after 20 epochs. The close matching between training and validation accuracies, together with control loss values, highlights effective convergence and generalization.

TABLE IV

MODELS PERFORMANCE METRICS BY TRAINING AND VALIDATION

Metric	Training	Validation
Accuracy	96%	94%
Loss	<0.2	~ 0.3
Epochs	20	20

F. CONCLUSION

This research performed a comparative analysis of two hybrid DL models, LSTM-CNN and LSTM-DNN, in terms of intrusion detection, malware classification, and anomaly detection of various cybersecurity datasets. The main objective was to evaluate their capability to estimate a temporal dependence and obtain discriminative features during uniform experimentation. The empirical findings proposed that the hybrid models both have strong detection performance, however, the LSTM-CNN beats LSTM-DNN in all the tasks considered. Temporal encoding is followed by the addition of convolutional layers to allow the LSTM- CNN to learn complementary spatial configurations, thus achieving increased mean accuracy, precision, recall, and F1-score, and less variability across repeated runs. Training and validation analyses also indicate the constant convergence and good generalization of training in a few epochs. Heat-map confusion matrix and convergence curve analyses of the above findings support these results through lower misclassification rates and increased robustness of LSTM-CNN. Although the investigation did not present a new architecture, it provided empirical support, that is, a practical benefit of temporal - spatial hybridization over the strictly hierarchical dense modelling of the cybersecurity applications of interest. Future studies would scale the analysis to bigger and more diverse real dataset, compute a rigorous statistical validation, and examine implementation-friendly improvements, such as model performance and explain-ability thus, aiding real-time security activities.

Author Contribution

Abdul Waheed Ahmad: conceptualization, investigation, writing– original draft, writing– review & editing. **Sadia Akbar:** conceptualization, investigation, writing– original draft. **Muhammad Adnan:**

conceptualization, investigation, writing–review & editing, supervision. **Jamal ud Din:** writing–review & editing

Conflict of Interest

The authors of the manuscript have no financial or non-financial conflict of interest in the subject matter or materials discussed in this manuscript.

Data Availability Statement

Data supporting the findings of this study will be made available by the corresponding author upon request.

Funding Details

No funding has been received for this research.

Generative AI Disclosure Statement

The authors did not use any type of generative artificial intelligence software for this research.

REFERENCES

[1] X. Li, H. Huang, G. Yuan, Z. Wang, and R. Du, "An intrusion detection method based on fusion neural network," *Front. Comput. Intell. Syst.*, vol. 4, no. 2, pp. 124–130, Jun. 2023, https://doi.org/10.54097/fcis.v4i2.10369.

[2] J. Bi, X. Zhang, H. Yuan, J. Zhang, and M. C. Zhou, "A hybrid prediction method for realistic network traffic with temporal convolutional network and LSTM," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 3, pp. 1869–1879, Jul. 2022, https://doi.org/10.1109/TASE.2021.3077537.

[3] J. Suganthi, B. Nagarajan, and S. Muhtumari, "Network anomaly detection using hybrid deep learning technique," in *Algorithms, Tools and Paradigms*, D. J. Hemanth *et al*., Eds., vol. 39, *Advances in Parallel Computing*. Amsterdam, The Netherlands: IOS Press, 2022, pp. 103–109, https://doi.org/10.3233/APC220014.

[4] X. Wan, H. Liu, H. Xu, and X. Zhang, "Network traffic prediction based on LSTM and transfer learning," *IEEE Access*, vol. 10, pp. 86181–86190, 2022, https://doi.org/10.1109/ACCESS.2022.3199372.

[5] A. Djenna, A. Bouridane, S. Rubab, and I. M. Marou, "Artificial intelligence-based malware detection, analysis, and mitigation," *Symmetry*, vol. 15, no. 3, art. no. 677, Mar. 2023, https://doi.org/10.3390/sym15030677.

[6] S. Morgan, "Cybercrime to cost the world $10.5 trillion annually by 2025," *Cybercrime Magazine*, Nov. 13, 2020. [Online]. Available: https://cybersecurityventures.com/hackerpocalypse-cybercrime-report-2016/. [Accessed: Feb. 25, 2025].

[7] M. Saied, S. Guirguis, and M. Madbouly, "Review of filtering-based feature selection for botnet detection in the Internet of Things," *Artif. Intell. Rev.*, vol. 58, no. 4, art. no. 119, Apr. 2025, https://doi.org/10.1007/s10462-025-11113-0.

[8] M. Ozkan-Okay *et al*., "A comprehensive survey: Evaluating the efficiency of artificial intelligence and machine learning techniques on cyber security solutions," *IEEE Access*, vol. 12, pp. 12229–12256, 2024, https://doi.org/10.1109/ACCESS.2024.3355547.

[9] J. W. Goodell and S. Corbet, "Commodity market interactions with energy-firm distress: Evidence from the Colonial Pipeline ransomware attack," *SSRN Electron. J.*, Jun. 2022, https://doi.org/10.2139/ssrn.4130544.

[10] D. E. Whitehead, K. Owens, D. Gammel, and J. Smith, "Ukraine cyber-induced power outage: Analysis and practical mitigation strategies," in *Proc. 70th Annu. Conf. Protective Relay Eng. (CPRE)*, College Station,

TX, USA, 2017, pp. 1–8, https://doi.org/10.1109/CPRE.2017.8090056.

[11] A. Greenberg, "The untold story of NotPetya, the most devastating cyberattack in history," *Wired*, Aug. 22, 2018. [Online]. Available: https://www.wired.com/story/notpetya-cyberattack-ukraine-russia-code-crashed-the-world/. [Accessed: Feb. 25, 2025].

[12] M. S. Akhtar and T. Feng, "Detection of malware by deep learning as CNN-LSTM machine learning techniques in real time," *Symmetry*, vol. 14, no. 11, art. no. 2308, Nov. 2022, https://doi.org/10.3390/sym14112308.

[13] S. Ali *et al*., "A novel approach of botnet detection using hybrid deep learning for enhancing security in IoT networks," *Alexandria Eng. J.*, vol. 103, pp. 88–97, Sep. 2024, https://doi.org/10.1016/j.aej.2024.05.113.

[14] M. Z. Alom *et al*., "A state-of-the-art survey on deep learning theory and architectures," *Electronics*, vol. 8, no. 3, art. no. 292, Mar. 2019, https://doi.org/10.3390/electronics8030292.

[15] M. Alauthman, N. Aslam, M. Alkasassbeh, S. Khan, A. Al-Qerem, and K.-K. R. Choo, "An efficient reinforcement learning-based botnet detection approach," *J. Netw. Comput. Appl.*, vol. 150, art. no. 102479, Jan. 2020, https://doi.org/10.1016/j.jnca.2019.102479.

[16] J. Saxe and K. Berlin, "Deep neural network-based malware detection using two-dimensional binary program features," in *Proc. 10th Int. Conf. Malicious Unwanted Softw. (MALWARE)*, Fajardo, PR, USA, 2015, pp. 11–20, https://doi.org/10.1109/MALWARE.2015.7413680.

[17] V. T. Patil and S. S. Deore, "DDoS attack detection: Strategies, techniques, and future directions," *J. Electr. Syst.*, vol. 20, no. 9s, pp. 2030–2046, Jul. 2024, https://doi.org/10.52783/jes.4808.

[18] I. A. Abdulmajeed and I. M. Husien, "MLIDS22—IDS design by applying hybrid CNN-LSTM model on mixed datasets," *Informatica*, vol. 46, no. 8, pp. 121–134, Nov. 2022, https://doi.org/10.31449/inf.v46i8.4348.

[19] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, https://doi.org/10.1038/nature14539.

[20] S. Han, H. Yun, and Y. Park, "Deep learning for cybersecurity classification: Utilizing depth-wise CNN and attention mechanism on VM-obfuscated data," *Electronics*, vol. 13, no. 17, art. no. 3393, Sep. 2024, https://doi.org/10.3390/electronics13173393.

[21] L. Nataraj, S. Karthikeyan, G. Jacob, and B. S. Manjunath, "Malware images: Visualization and automatic classification," in *Proc. 8th Int. Symp. Visualization Cyber Security (VizSec '11)*, Pittsburgh, PA, USA, 2011, art. no. 4, pp. 1–7, https://doi.org/10.1145/2016904.2016908.

[22] D. Kilichev, D. Turimov, and W. Kim, "Next-generation intrusion detection for IoT EVCS: Integrating CNN, LSTM, and GRU models," *Mathematics*, vol. 12, no. 4, art. no. 571, Feb. 2024, https://doi.org/10.3390/math12040571

[23] H. Kim and M. Kim, "Malware detection and classification system based on CNN-BiLSTM," *Electronics*, vol. 13, no. 13, art. no. 2539, Jul. 2024, https://doi.org/10.3390/electronics13132539.

[24] B. Mohammed and E. Gbashi, "Intrusion detection system for NSL-

KDD dataset based on deep learning and recursive feature elimination," *Eng. Technol. J.*, vol. 39, no. 7, pp. 1069–1079, Jul. 2021, https://doi.org/10.30684/etj.v39i7.1695.

[25] S. Shende and S. Thorat, "Long short-term memory (LSTM) deep learning method for intrusion detection in network security," *Int. J. Eng. Res. Technol. (IJERT)*, vol. 9, no. 6, pp. 1–6, Jun. 2020, https://doi.org/10.17577/IJERTV9IS061016.

[26] F. E. Laghrissi, S. Douzi, K. Douzi, and B. Hssina, "Intrusion detection systems using long short-term memory (LSTM)," *J. Big Data*, vol. 8, no. 1, art. no. 65, May 2021, https://doi.org/10.1186/s40537-021-00448-4.

[27] D. S. Berman, A. L. Buczak, J. S. Chavis, and C. L. Corbett, "A survey of deep learning methods for cyber security," *Information*, vol. 10, no. 4, art. no. 122, Apr. 2019, https://doi.org/10.3390/info10040122.

[28] R. Baimukashev, K. Artykbayev, K. Adam, and B. Mels, "Intrusion detection system for wireless networks," in *Proc. 16th Int. Conf. Electron. Comput. Computation (ICECCO)*, Kaskelen, Kazakhstan, 2021, pp. 1–5, https://doi.org/10.1109/ICECCO53203.2021.9663787.

[29] Y. Li, S. Chai, Z. Ma, and G. Wang, "A hybrid deep learning framework for long-term traffic flow prediction," *IEEE Access*, vol. 9, pp. 11264–11271, 2021, https://doi.org/10.1109/ACCESS.2021.3050836.

[30] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, 2017, pp. 2261–2269, https://doi.org/10.1109/CVPR.2017.243.

[31] F. Aksan, Y. Li, V. Suresh, and P. Janik, "CNN-LSTM vs. LSTM-CNN to predict power flow direction: A case study of the high-voltage subnet of northeast Germany," *Sensors*, vol. 23, no. 2, art. no. 901, Jan. 2023, https://doi.org/10.3390/s23020901.

[32] S. Tipper, H. F. Atlam, and H. S. Lallie, "An investigation into the utilisation of CNN with LSTM for video deepfake detection," *Appl. Sci.*, vol. 14, no. 21, art. no. 9754, Nov. 2024, https://doi.org/10.3390/app14219754.

[33] Hassan06, "NSL-KDD," *Kaggle*. [Online]. Available: https://www.kaggle.com/datasets/hassan06/nslkdd. [Accessed: Jul. 28, 2026].

[34] Ernie55ernie, "Improved CICIDS2017 and CSE-CICIDS2018," *Kaggle*. [Online]. Available: https://www.kaggle.com/datasets/ernie55ernie/improved-cicids2017-and-csecicids2018. [Accessed: Jul. 28, 2026].

[35] Microsoft, "Microsoft Malware Classification Challenge (BIG 2015)," *Kaggle*. [Online]. Available: https://www.kaggle.com/c/malware-classification. [Accessed: Jul. 28, 2026].

[36] K. V. Nguyen, H. T. Nguyen, T. Q. Le, and Q. N. M. Truong, "Abnormal network packets identification using header information collected from Honeywall architecture," *J. Inf. Telecommun.*, vol. 7, no. 4, pp. 437–461, 2023, https://doi.org/10.1080/24751839.2023.2215135.

[37] Y. Zhu, S. Yao, and X. Sun, "Feature interaction dual self-attention network for sequential recommendation," *Front. Neurorobot.*, vol. 18, art. no. 1456192, Aug. 2024, https://doi.org/10.3389/fnbot.2024.1456192.