Computer Modeling in Engineering & Sciences

Tech Science Press

Doi:10.32604/cmes.2025.064926

ARTICLE

Dual-Channel Attention Deep Bidirectional Long Short Term Memory for Enhanced Malware Detection and Risk Mitigation

Madini O. Alassafi and Syed Hamid Hasan*

Department of Information Technology, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, 22254, Saudi Arabia

*Corresponding Author: Syed Hamid Hasan. Email: shhasan@kau.edu.sa

Received: 27 February 2025; Accepted: 15 July 2025; Published: 31 August 2025

ABSTRACT: Over the past few years, Malware attacks have become more and more widespread, posing threats to digital assets throughout the world. Although numerous methods have been developed to detect malicious attacks, these malware detection techniques need to be more efficient in detecting new and progressively sophisticated variants of malware. Therefore, the development of more advanced and accurate techniques is necessary for malware detection. This paper introduces a comprehensive Dual-Channel Attention Deep Bidirectional Long Short-Term Memory (DCA-DBiLSTM) model for malware detection and risk mitigation. The Dual Channel Attention (DCA) mechanism improves the model's capability to concentrate on the features that are most appropriate in the input data, which reduces the false favourable rates. The Bidirectional Long, Short-Term Memory framework helps capture crucial interdependence from past and future circumstances, which is essential for enhancing the model's understanding of malware behaviour. As soon as malware is detected, the risk mitigation phase is implemented, which evaluates the severity of each threat and helps mitigate threats earlier. The outcomes of the method demonstrate better accuracy of 98.96%, which outperforms traditional models. It indicates the method detects and mitigates several kinds of malware threats, thereby providing a proactive defence mechanism against the emerging challenges in cybersecurity.

KEYWORDS: Cybersecurity; risk mitigation; malware detection; bidirectional long short-term memory; dual-channel attention

1 Introduction

The internet's rapid growth is changing people's lifestyles, allowing social life to blend with online life, which increases security threats in a variety of ways for both individuals and organizations [1]. Cyberattacks are crimes that aim at networks or computers, which involve altering data, stealing personal details, and interrupting processes in computer networks and other electronic systems to cause problems [2]. The effects of a successful attack go beyond mere disruption; they produce threats to a nation's security, economic stability, and citizens' welfare [3]. This has led to the emergence of cybersecurity, which is an essential aspect for almost every kind of individual, corporation, and government to safeguard and keep their data private and increase their businesses. Cybersecurity is the process of applying cyber protective measures and policies to defend network infrastructures, servers, programs, and data from attacks, unauthorized access, or changes [4]. Malware attacks continue to become widespread and ever-changing, producing threats to digital security as attackers continue to create new methods to circumvent existing security measures [5].

Cyber attacks, such as financial losses, unauthorized exposure of data, and reputational damage, highlight the urgent requirement for strong cybersecurity policies.

Traditional cybersecurity strategies struggle to detect evolving new threats requiring recurrent updates and are helpful only for short-term time dependency predictions [6]. These techniques' lack of ability to learn dynamically from new data renders them inefficient in dealing with new attack tactics such as obfuscated malware, zero-day exploits, and sophisticated persistent threats [7]. Therefore, highly resilient and flexible detection systems that can detect unknown or disguised threats are badly needed. Consequently, for threat identification and safeguarding, traditional cybersecurity solutions are advancing to sophisticated deep and machine learning-based systems [8]. These approaches can detect both known and new threats and attacks with a high degree of accuracy and a reduced false positive rate. They also guarantee data accessibility, security, and integrity [9].

Even though there has been remarkable progress using various machine learning and deep learning methods in detecting threats, these approaches still need to improve, given the inability to adapt to evolving threats. These methods also tend to generate a high number of false positives; in real-time, they are also slow in data analysis [10]. To address these limitations in the current models, this paper proposes a new DCA-DBiLSTM framework for the analysis. This model combines Bidirectional Long Short-Term Memory (BiLSTM) for sequence modelling with DCA to address the problem of limited context. BiLSTM, due to its bidirectional nature, can capture long-term dependencies in the sequential data, and DCA can enhance the feature extraction by attending to both channel and spatial relevance, which enhances the overall detection of malware and management of risk.

Contribution

The contribution of the paper is as follows:

***Dual-Channel Attention Mechanism for Malware Detection*:** A new DCA-DBiLSTM model that integrates a BiLSTM layer and a DCA layer is proposed. The bidirectional Long Short-Term Memory (LSTM) extracts intricate features from the input data by incorporating both past and future contextual information. The DCA mechanism improves the feature selection process and contributes to the decrease of the false positive rate. Hence, this integration enables the model to identify malware with high levels of efficiency.

Comprehensive Malware Classification and Risk Mitigation Framework: The DCA-DBiLSTM framework proposed in this paper has two phases, malware detection and risk mitigation. This phase determines the level of impact of the malware and is able to reduce the impact of the threats at an early stage. This method offers a preventive strategy and is therefore a major contribution to the cybersecurity field.

Application to Real-World Datasets: The model is evaluated through different experiments, and its applicability on different types of malware datasets is proved with the help of two real-world datasets, CICAndMal2017 and Android Malware Detection. The results proved that the proposed model can effectively identify and prevent several types of malware attacks. It is also capable of handling large data sets, which is very useful in real-life situations.

The rest of the paper is organised as follows: Section 2 discusses the available methods for identifying malware and preventing threats by employing machine learning and deep learning. The detailed description of the proposed DCA-DBiLSTM technique is presented in Section 3, which explains the integration of Bidirectional LSTM and DCA. The results of experiments, which were carried out in order to evaluate the applicability of the proposed model, are given in Section 4, where several indices and comparisons are depicted. Last but not least, Section 5 provides the conclusion of the paper and the future work.

2 Literature Survey

In this section, the current approaches to the detection of malware and its associated risk that have been adopted will be discussed alongside several approaches to detection and classification using machine learning and deep learning methods.

Yazdinejad et al. [11] proposed a secure intelligent fuzzy blockchain system via Deep Learning (DL), designed to determine the presence of threats within an IoT system that interacts with blockchain technology. For threat identification within the networks, this model used an optimised adaptive neuro-fuzzy inference system (ANFIS). The model was very efficient in identifying threats in the blockchain Internet of Thing (IoT) framework, as far as throughput and latency are concerned. In the previous work, Bhandari et al. [12] introduced the Distributed Deep Neural Network-Based Middleware to detect cyberattacks in a smart IoT ecosystem. The scheme in this model was an Artificial Intelligence (AI)-based one that identified malware and attacks on IoT devices and also constantly analysed network traffic data to prevent possible threats in bright conditions. The results showed that the model was a very good one at identifying threats and malware in new settings while using very little power and memory.

Baldini [13] developed a Recurrence Quantification Analysis (RQA) based model to find cybersecurity spoofing attacks in vehicular networks. This model employed a sliding window method, which helped reduce the dimension relative to the window size. Also, the processing of the Controller Area Network (CAN) bus messages was not required. The RQA-based method demonstrated superior performance and better accuracy in detecting cyberattacks. Rajak and Tripathi [14] presented a 5G-based system for the identification and classification of cyber-attacks that utilized a Deep Learning-Stacked Long Short-Term Memory (DL-SkLSTM) architecture. The DL-SkLSTM framework distinguished several categories of cyberattacks accurately. This model very excellently identified and classified several kinds of cyberattacks within the IoT environments.

Jiang et al. [15] presented a Blockchain and Federated Learning for Sharing (BFLS) threat detection models as Cyber Threat Intelligence (CTI) to protect data confidentiality. The Federated Learning (FL) trained the BFLS model for threat detection, while the blockchain shared information to overcome hostile nodes and failing servers. Through BFLS deployment, the CTI was effectively secured and had excellent threat detection efficiency.

Dey et al. [16] presented Gravitational Search Grey Wolf-Decision Tree for threat detection. In particular, ensemble learning-based classifiers were used for threat detection and metaheuristic-based ensemble feature selection for optimal feature selection. The model demonstrated excellent performance as it accurately identified cyber threats and also minimized false positives. Vijayalakshmi and Karthika [17] presented a hybrid dual-channel convolution neural network with spider monkey optimization (DCCNN-SMO) to detect files that contain malware and pirated software within the IoT networks. A method for detecting software piracy was employed to find the copied original software's characteristics. The DCCNN-SMO approach was highly effective and also helped retrieve the most accurate categorization information.

Research Gap and Limitations

The existing methods in cybersecurity for detecting malware struggle to detect newer threats that are evolving day by day, which most often lead to high false positive rates and could be more computationally efficient in the real-time processing of data. Data from the past and patterns in sequential data are used, which are useful only for short-term time dependency predictions and also have a lower accuracy in threat detection. To overcome these limitations, this paper introduces a novel Dual-Channel Attention Deep Bidirectional Long Short-Term Memory (DCA-DBiLSTM) model. This model is highly adaptable, although there are a variety of variabilities in different datasets, as it is capable of grasping complex patterns that are

relevant to both benign and malicious applications. A risk mitigation phase is also introduced, along with malware detection, that helps in the systematic mitigation of possible threats and vulnerabilities by assessing the threats' severity and their potential effects. This integration of DCA with BiLSTM effectively detects both known and new attacks with a higher accuracy rate and lower computation time. Also, it helps effectively mitigate threats earlier.

3 Proposed Methodology

Fig. 1 presents the Dual-Channel Attention Deep Bidirectional Long Short-Term Memory (DCA-DBiLSTM) model for malware detection and risk mitigation. Initially, two high-quality datasets were collected, namely, CICAndMal2017 and the Android Malware Detection dataset, for the malware detection process. Then, the preprocessing steps, including data cleaning, filtering, normalisation, and transformation, are carried out to make sure the images are of high quality. Then, model training takes place, where the training data, i.e., 80% of the total images, are passed into the DCA-DBiLSTM model for training. The BiLSTM framework enables forward and backward processing of the data. This helps the model to grasp patterns in malware behaviour. Finally, malware is detected if present, and it is analysed to detect its risks. If there are potential risks that need to be mitigated before any further damage happens, appropriate strategies are executed to protect systems.

3.1 Data Collection

Data collection is a crucial step that involves collecting data from various sources to analyse the model for detecting malware.

CICAndMal2017 Dataset: This research utilises a sophisticated malware dataset. It can be retrieved from https://www.unb.ca/cic/datasets/andmal2017.html (accessed on 01 January 2025). The Canadian Institute for Cybersecurity provided this dataset to train effective malware detection classifiers. CICAndMal2017 encompasses more than 10,854 samples from various origins, including 4354 samples for malware and 6500 samples for benign. This dataset consists of five types of contemporary malware, such as adware, rootkit, ransomware, scareware, and Short Messaging Service (SMS) Malware, and one benign file, which contains three types of benign samples, including benign2015, benign2016, and benign2017, in three separate folders. Each malware category is made up of various malware families, where each family contains a collection of data instances. The dataset includes 85 labelled features (characteristics) as well, which are described in the feed network traffic. The details of this dataset are shown in Table 1.

Android Malware Detection Dataset: The Android Malware Detection dataset is retrieved from https://www.kaggle.com/datasets/subhajournal/android-malware-detection (accessed on 01 January 2025). It is developed to help detect and classify malicious apps on Android devices. This dataset usually consists of a range of attributes such as Application Programming Interface (API) requests, permissions, and other static and dynamic features extracted from applications.

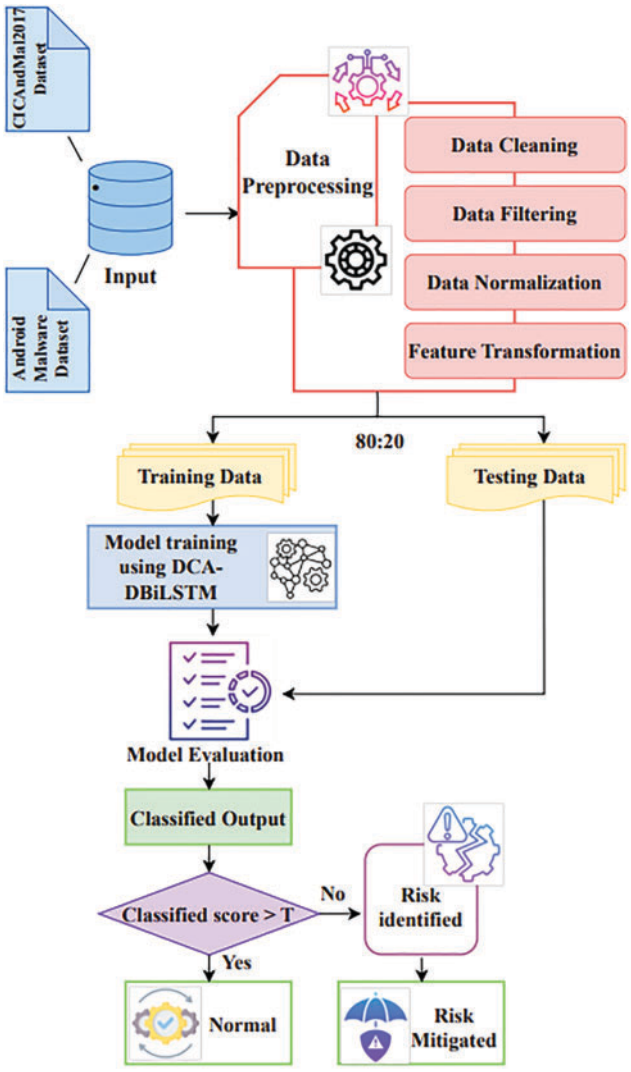

Figure 1: Conceptual framework of the proposed methodology

Table 1: Dataset details

Folders	Number of folders	Number of samples
Adware	10	104
Rootkit	10	105
Ransomware	10	101
Scareware	11	112
SMS Malware	11	109
Benign	3	1702

Total Samples: Around 10,000 to 20,000 samples (depending on the particular dataset chosen the precise count may differ).

Classes: There are different classes in each dataset, as there are numerous kinds of malware, like adware, ransomware, spyware, etc. Each of these contributes to different classes depending on their nature and their impacts on networks. Table 2 provides the dataset's class distribution.

Table 2: Class distribution

Class	Number of samples
Benign	7000
Ransomware	2000
Spyware	2500
Adware	3000
Trojan	2500
Potentially Harmful Apps (PHA)	3000

Table 3: Parameter settings of the DCA-DBiLSTM method

Parameter	Values
Learning rate	0.001
Batch size	64
Epochs	100
Dropout rate	0.4
Activation function	ReLU
Spatial attention	3 × 3 Kernel size
Hidden layers	3
Loss function	Cross-entropy

3.2 Data Preprocessing Phase

This is an important step in preparing datasets, which involves data cleaning, data filtering, data normalization, and feature transformation to make it suitable for analysis.

3.2.1 Data Cleaning

Data cleaning is the process of correcting errors in the data. It also involves deleting the data if it is found to be inaccurate and removing duplicate data. This helps maintain high-quality data so as to make more accurate decisions.

3.2.2 Data Filtering

Data filtering involves limiting the amount of data for training by removing unwanted data that is irrelevant or of poor quality. This process eliminates the outliers as these affect the accuracy of the model. Thus, the filtering process filters out only the relevant data that is required.

3.2.3 Data Normalization

To enhance the malware detection model's performance, data normalization is used to transform the dataset's features to ensure they fall on a common scale. Normalization of the data is particularly important

as features with different scales cause distortions. The data in our dataset is transformed so that the mean becomes zero and the standard deviation becomes one, so that no single feature affects the training process excessively. The normalization of each feature y to a scale of $[0, 1]$ is given below.

$$y' = \frac{y - \min(y)}{\max(y) - \min(y)} \tag{1}$$

3.2.4 Feature Transformation

Feature transformation is done to enhance the performance of the malware detection model, where the datasets' original features are changed to make them more suitable for the model's training. This process reduces the data's dimensions and makes the dataset simpler without eliminating essential details, so that the data becomes more manageable for the model.

For a categorical feature F with l categories, create l binary features:

$$F' = \begin{cases} 1 & if\ category\ is\ present \\ 0 & otherwise \end{cases} \tag{2}$$

3.3 Malware Detection Using a Novel Dual Channel Attention Deep Bidirectional Long Short Term Memory (DCA-DBiLSTM) Model

Along with the rapid expansion of the internet, malware attacks are also increasingly evolving day by day, which demands the urgent need for more and more malware detection systems. Conventional malware detection models need help to detect new malware due to their changing nature and high false positive rates. Therefore, a sophisticated deep-learning model is required to detect these malwares that are evolving every day accurately. Consequently, the DCA-DBiLSTM model is developed in this paper, which integrates the potential of the BiLSTM framework with the DCA mechanism to detect malware and mitigate risks as early as possible. The dual-channel attention mechanism improves the model's ability to concentrate on the most apprdata'se features of the input data, which reduces the false positive rates. The BiLSTM framework processes the data in both forward and backward directions. It assists in capturing some of the important relationships that exist between past and future events, which are valuable in improving the malware behaviour model. The model can handle sequences of data and is scalable to large data sets since it is invariant to variability in different datasets.

3.3.1 DCA-DBiLSTM Model

The BiLSTM framework takes the data in forward as well as backward directions. It also assists in capturing the most important dependency from previous and subsequent scenarios, which is valuable for the model to learn more about the behaviours of malware. It makes the model better in understanding the malware activity dynamics as opposed to the initial model.

Forward LSTM: The input sequence is processed from start to finish by the forward LSTM, capturing temporal reliance and subtlety in the data, thereby learning to recognize patterns representative of malware based on the feature's sequential arrangement derived from software behaviours.

Forget Gate: From the previous cell state, which information needs to be discarded is decided by this gate. The forget gate is computed as shown below:

The forward *LSTM* processes the input sequence $Q = [q_1, q_2, \ldots, q_S]$:

$$k_r^g = LSTM\left(q_r, k_{r-1}^g\right) \tag{3}$$

Input Gate: This gate manages the new data addition to the state of the cell. It is calculated as shown below:

$$g_r = \sigma\left(X_g \cdot \left[k_{r-1}^g, q_r\right] + a_g\right) \tag{4}$$

Cell StateUpdate: Initially, a candidate cell state is generated to update the state of the cell as shown below:

$$\tilde{F}_r = \tanh\left(X_F \cdot \left[k_{r-1}^g, q_r\right] + a_F\right) \tag{5}$$

$$F_r = g_r \cdot F_{r-1} + j_r \cdot \tilde{F}_r \tag{6}$$

Output Gate: Utilizing the current state of the cell, this gate decides the output for the present time step as expressed below:

$$k_r = \left[k_r^g; k_r^a\right] \tag{7}$$

Backward LSTM: This element processes the input sequence in the reverse direction, i.e., from end to start. This enables the model to acquire additional context that improves the understanding of the model. It also allows the DB-LSTM to understand how future actions can notify the elucidation of past behaviours, which is necessary for the precise identification of malware.

The backward *LSTM* processes the sequence in reverse:

$$k_r^a = LSTM\left(q_{S-r+1}, k_{r-1}^a\right) \tag{8}$$

Combined BiLSTMOutput: The outputs of both the forward and backward LSTMs are merged, which enables the model to obtain details from both directions. This comprehensive output improves the model's grasp of input data and thereby enhances its capability to identify subtle signs of malicious activity.

The outputs from both LSTMs are combined:

$$k_r = \left[k_r^g; k_r^a\right] \tag{9}$$

3.3.2 Dual Channel Attention Mechanism

The fusion of the Spatial Attention Module (SAM) and Channel Attention Module (CAM) improves the efficiency of computation. It substantially enhances the performance of the model in the identification of malware. The processing data's dimensionality is minimized by the DCA mechanism by employing targeted attention to the feature space, allowing the model to concentrate on the most informative characteristics. In cybersecurity tasks, where specific characteristics of malware may be more pertinent than others, this adaptive refinement of features is crucial. The combined use of these attention mechanisms leads to a more robust and versatile model that can achieve excellent performance in the detection of different malware types. Additionally, global and local feature aggregation techniques, such as max pooling and average pooling, help retain both high-level and fine-grained details. The model is trained with diverse and evolving malware datasets to generalize well to unseen threats. Furthermore, regularization techniques like dropout and batch normalization prevent the model from over-relying on dominant features, ensuring it remains sensitive to new and subtle variations in malware behavior. The DCA mechanism's architecture is illustrated in Fig. 2.

- ***Channel Attention Module (CAM)***

By learning weights for each channel based on their contribution to the process of classification, the CAM highlights the compliance of various feature channels. This prioritization enables the network to concentrate on features that are most important in differentiating between malicious and benign applications. To compute a channel attention map, CAM works by integrating global information from the feature map G. The steps are expressed mathematically as:

$$N_F = \sigma\left(MLP\left(Global\ \text{Avg}\ Pool\ (\text{G})\right) + MLP\left(Global\ \text{Max}\ Pool\ (\text{G})\right)\right) \tag{10}$$

here, MLP is a multi-layer perceptron, $Global$ Avg $Pool$ and $Global$ Max $Pool$ summarize the feature maps along the spatial dimensions, and the sigmoid activation function is σ. The output attention map N_F is then multiplied by the input feature map G:

$$G_F = N_F \odot G \tag{11}$$

This multiplication represses the less important channel features and improves the significant ones.

- ***Spatial Attention Module (SAM)***

SAM is important for improving the effectiveness of deep learning models, especially in malware identification tasks where localized characteristics are essential for recognizing malicious patterns. CAM efficaciously picks out and emphasizes the key features over different channels, whereas SAM focuses on particular spatial areas of the input data, like the behaviour patterns or applications' features.

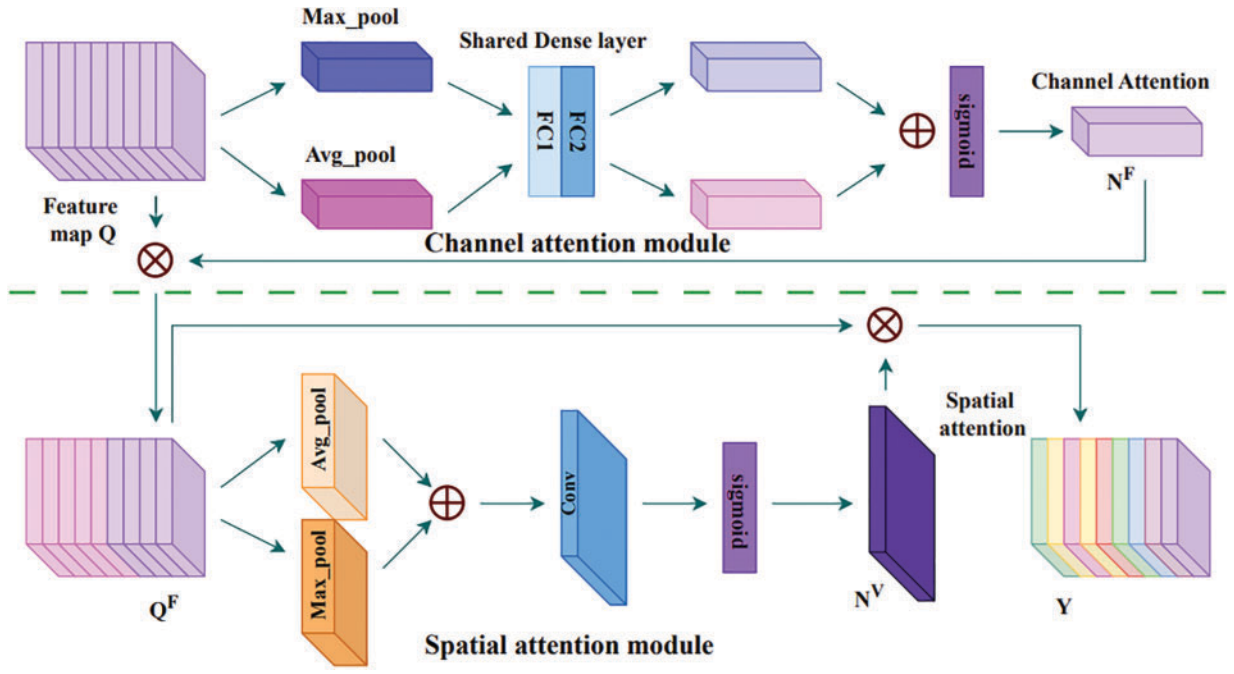

Figure 2: Dual channel attention mechanism

SAM evaluates the significance of various regions within the feature map by creating a spatial attention map. This makes it possible for the model to give emphasis on the most important segments that can facilitate accurate malware differentiation. Such spatial refinement allows the network to be more sensitive to localized

patterns, which is especially crucial when small variations in the behavior of the application or code structure signal malicious activity.

However, when trying to extract spatial information from the feature map enhanced by CAM, SAM uses average maximum pooling and normal average pooling. This process produces spatial attention scores for every location in the feature map in order to indicate which area is more important to the detection task. As soon as the spatial attention map is established, it is utilized to modify the feature map, emphasizing more important areas while minimizing the impact of less informative regions. Fig. 3 displays the DCA-BiLSTM model's architecture.

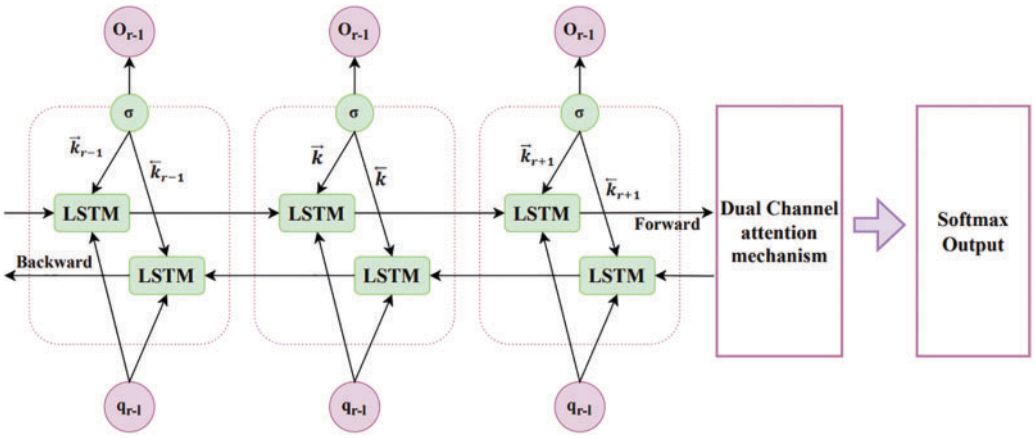

Figure 3: Architecture of DCA-BiLSTM model

This concentrated emphasis on spatial relationships makes sure that the model captures subtle details in the input data that are otherwise overlooked. For example, it enables the detection system to focus on specific behaviours or features of applications that indicate possible threats. Hence, this leads to enhanced representation of features and increased accuracy in classification, allowing the model to distinguish between benign applications and different kinds of malware more effectively.

$$N_V = \sigma\left(Conv\left(\text{Avg}\,Pool\left(\text{G}_F\right), \text{Max}\,Pool\left(\text{G}_F\right)\right)\right) \tag{12}$$

here, $Conv$ represents a convolutional operation. Similar to CAM, the spatial attention map N_V is multiplied by the channel-refined feature map:

$$G_V = N_V \odot G_F \tag{13}$$

Final Classification: The aggregated results are then inputted along with a softmax activation function into a dense layer for classification which is represented as follows:

$$x = soft\max\left(\text{X} \cdot k_r + a\right) \tag{14}$$

here, the output vector is x which represents the probability distribution over the classes (e.g., benign or different kinds of malware).

3.4 Risk Mitigation Phase

The risk mitigation stage in cybersecurity includes systematically mitigating possible threats and vulnerabilities after detecting and evaluating them. Risk mitigation starts with prioritizing risks based on their potentiality and consequence, followed by the execution of a mix of improvements in physical security, technical controls like firewalls, encryption, and access controls, and administrative strategies such as security measures and staff training. To reduce damage in the event of a security breach, it is essential to monitor systems for irregularities and plan effective incident response continuously. In the context of malware identification, the risk mitigation step is a critical step following the evaluation of the model and the possible detection of malware threats. To safeguard computers and networks from vulnerability, it concentrates on the practical application of the insights obtained from the predictions of the model. The risk mitigation phase usually works as described below:

- ***Malware Classification and Flagging***

It is utilized to categorize and detect malware within the dataset following the training and assessment of the DCA-DBiLSTM. As the model processes the test data, it labels the samples and classifies them as malicious or benign. The samples that are classified as benign are verified as safe. However, samples that are classified as malicious are flagged because they are possible threats. These malware outputs are usually categorized by their types, such as adware, ransomware, spyware, etc.

- ***Risk Assessment***

Once malware threats are detected, the subsequent step is to evaluate their possible effect on the system. This includes assessing threats' severity and possible impacts, where factors like the malware type, its targeted system, and the possible damage it can cause are all assessed. This aids in prioritizing which threats require immediate attention and which can be addressed later. The pseudocode of the Risk Mitigation and Risk assessment is illustrated in Algorithm 1.

Algorithm 1: Risk mitigation and risk assessment

Input: testing_data, model, malware_types, risk_criteria
Output: flagged_malware, risk_assessment, mitigation_actions
Initialize empty list: flagged_malware = [], risk_assessment = [], mitigation_actions = []
For each sample in testing_data:
    Predict label using model (malicious or benign)
    If the label is 'malicious':
        Add sample to flagged_malware list
        For each malware sample in flagged_malware:
            Assess risk_level based on malware type (e.g., ransomware, spyware), system target, and potential damage
            Append risk assessment to risk_assessment list
            If risk_level == 'High':
                Execute isolation action, apply patches, and notify security teams
            Else if risk_level == 'Medium':
                Monitor behavior, apply partial isolation
            Else:
                Log malware info and continue monitoring
Return flagged_malware, risk_assessment, mitigation_actions

4 Experimental Results

This section discusses the evaluation metrics used to evaluate the DCA-DBiLSTM method's effectiveness. It also presents in detail the experimental configuration and the parameter settings needed for training and evaluating the DCA-DBiLSTM model. The results of the complete analysis of the model to show its efficiency are also presented here in this section.

4.1 Experimental Configuration

The proposed DCA-DBiLSTM model is implemented on Google Colab Pro where the model can be trained and tested at high speed, which is important for model tuning and comparison. This research's experimental setup is as follows: The language used in programming is Python. The DCA-DBiLSTM model is developed using TensorFlow and Keras library to train the proposed approach. For the experiments in this paper, the Google Colab Pro is used with a Tesla T4 GPU and 25 GB of RAM.

4.2 Parameter Settings

Several hyperparameters were fine-tuned by empirical experimentation to achieve the most effective performance in detecting malware. These hyperparameters include learning rate, batch size, epochs, dropout rate, activation function, spatial attention, hidden layers, and loss function.

4.3 Evaluation Measures

This section presents a detailed description of the measures used for evaluating the performance of the DCA-DBiLSTM model. Mainly, the research community used the Accuracy, Precision, Recall, and F1 scores in the testing process of the test dataset concerning the classification model, object recognition in machine learning, deep learning, and retrieval of information.

Accuracy **(***Acc***):** Accuracy is the overall proportion of correctly identified instances i.e., correctly identified instances' count divided by the total predictions' count in the dataset. Accuracy is mathematically defined as follows:

$$Acc = \frac{Y_p + Y_m}{Y_p + Y_m + N_p + N_m} \tag{15}$$

here, Y_p is true positive, indicating positive instances predicted as positive; N_p is false positive, which refers to negative data predicted as positive. Y_m is true negative, indicating negative instances predicted as negative and N_m is false negative, which refers to positive data predicted as negative.

Precision **(***pre***):** Precision is the proportion of actual positives correctly predicted by the model, i.e., true positives' count divided by the total positive predictions' count. Mathematically, it is calculated as:

$$pre = \frac{Y_p}{Y_p + N_p} \tag{16}$$

Recall **(**R_C**):** Recall, also known as sensitivity or true positive rate (TPR) is the proportion of true positives that are correctly identified by the model, i.e., the true positives' count divided by the total positive samples' count. Mathematically it is represented as:

$$R_C = \frac{Y_p}{Y_p + N_m} \tag{17}$$

F1 score **(*F*1):** It is a measure of performance for classification, which is calculated as the precision and recall harmonic mean. *F*1 score is expressed mathematically as:

$$F1 = \frac{2 \times (pre \times R_C)}{pre + R_C} \tag{18}$$

4.4 Performance Analysis

This section discusses the DCA-DBiLSTM model's performance analysis, which is used to assess its performance based on various evaluation measures. This analysis helps identify the effectiveness of the model in precise malware detection, which helps in earlier risk mitigation.

Fig. 4 displays two confusion matrices: (a) for the CICAndMal2017 dataset and (b) for the Android malware detection dataset. These matrices demonstrate the efficiency of the model in classifying several types of attacks within the datasets. Each row depicts the actual attack type, and each column depicts the predicted attack type. The correctly identified instances are represented as the diagonal elements, and incorrect predictions are described as the off-diagonal elements. Fig. 4a demonstrates that the model achieves accuracies of 98.89% for Adware, 99.03% for Ransomware, 99.11% for Scareware, and 99.37% for SMS Malware classes of the CICAndMal2017 dataset. Fig. 4b demonstrates that the model achieves accuracies of 98.59% for Adware, 98.73% for Scareware, 98.95% for SMS Malware, and 99.01% for Benign classes of the Android malware detection dataset. These high accuracies show the effectiveness of the model's predictions in classifying cyberattacks.

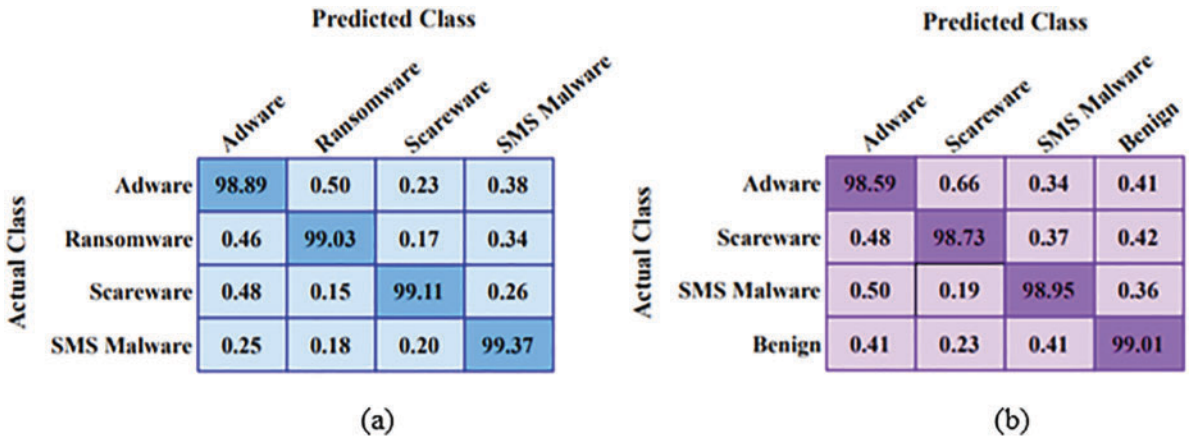

Figure 4: Confusion matrix for (**a**) CICAndMal2017 and (**b**) Android malware detection datasets

Fig. 5a illustrates the training and testing accuracy curves for the DCA-DBiLSTM model. The training accuracy curve demonstrates that the model's accuracy increases with each epoch of the training data. The testing accuracy also increases optimally with the training accuracy. The accuracy of training and testing rises over the 10th epoch, achieving 0.96 and 0.94, respectively. Fig. 5b illustrates the training and testing loss curves for the DCA-DBiLSTM model. The decrease in loss as the training progresses indicates that the model learns effectively on the training dataset to make better predictions. The training and testing loss decreases over the 10th epoch, achieving 0.10 and 0.19, respectively. It is analyzed from the graph that the model not only learns well from the training data but also effectively generalizes to new, unseen data.

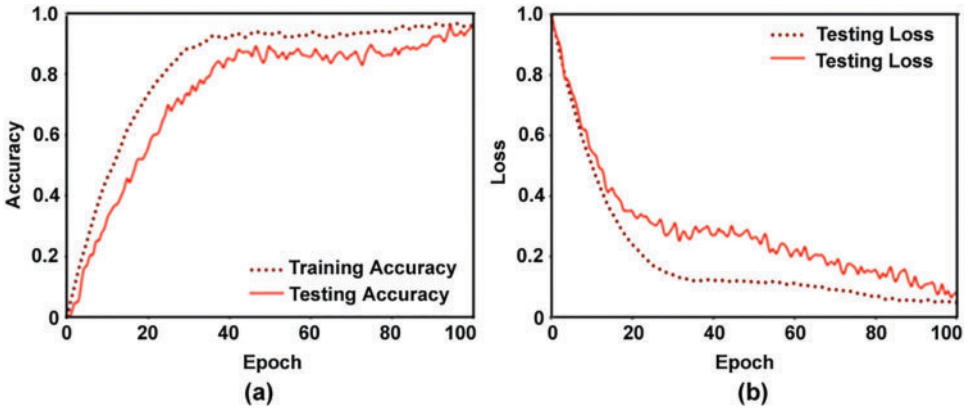

Figure 5: (**a**) Accuracy vs. Epoch graph and (**b**) Loss vs. Epoch graph

Table 4 presents the DCA-DBiLSTM's performance analysis using the CICAndMal2017 and the Android Malware Detection dataset for various evaluation measures such as accuracy, precision, recall, F1-score, and AUC-ROC. The higher values attained in all the metrics indicate better model performance, with the model achieving the highest performance on the CICAndMal2017 dataset.

Table 4: Performance analysis of datasets across various performance metrics

Measures	Datasets	
	CICAndMal2017	**Android malware detection**
Accuracy	99.10	98.82
Precision	98.43	98.25
Recall	98.71	98.67
F1 score	97.92	97.54
AUC-ROC	0.989	0.981

Fig. 6 displays the DCA-DBiLSTM precision-recall (PR) curve for the CICAndMal2017 dataset and the Android Malware Detection dataset. The PR curve represents the balance achieved between precision and recall at different ranges to assess the classification model's effectiveness in predicting different kinds of cyberattacks. A high value in precision signifies the model's effectiveness in correctly identifying every positive event. The high value in recall indicates that the DCA-DBiLSTM model is effective in detecting all correct data.

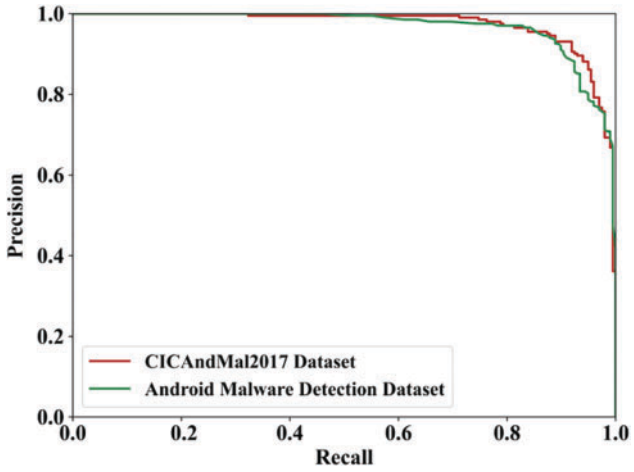

Figure 6: Precision-recall curve

4.5 Comparative Analysis

In this section, the DCA-DBiLSTM is compared with other state-of-the-art approaches, including Fuzzy DL [11], DL-SkLSTM [14], BFLS [15], Meta-ICTD [16], and DCCNN-SMO [17]. In the following part of the section, we compare different evaluation measures for the DCA-DBiLSTM and other existing methods as depicted in Fig. 7. Therefore, the present analysis reveals that the proposed DCA-DBiLSTM model has achieved the A, P, R, and F1 score values of 98.96%, 98.34%, 98.69%, and 97.73%, respectively, which is the best among all the models. From the results, the DCA-DBiLSTM is able to predict the threats with high accuracy, which then leads to proper risk management.

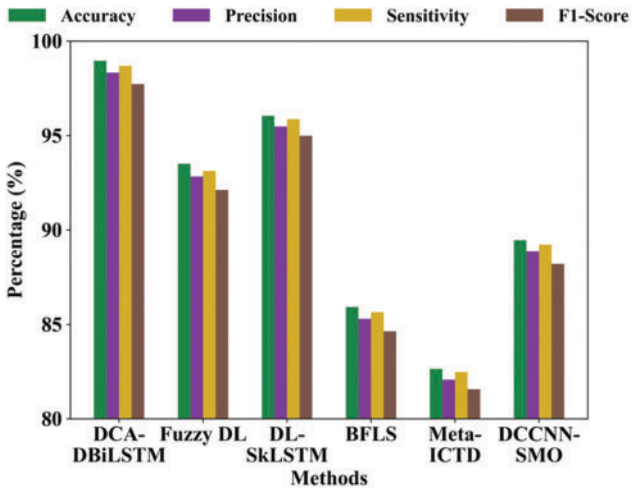

Figure 7: Comparative analysis of accuracy, precision, sensitivity, and F1-score

Based on Fig. 8, the comparison of the AUC-ROC evaluation of the DCA-DBiLSTM method with other existing methods is given. This curve shows the relationship between the rates of true positives and false positives in different result ranges. The DCA-DBiLSTM model attains a higher AUC-ROC value of 0.985 as compared to other existing models such as Fuzzy DL, DL-SkLSTM, BFLS, Meta-ICTD, and DCCNN-SMO,

which achieved values of 0.853, 0.908, 0.798, 0.787, and 0.820, respectively. The comparison results show that the DCA-DBiLSTM model has better performance with a high rate of true positives.

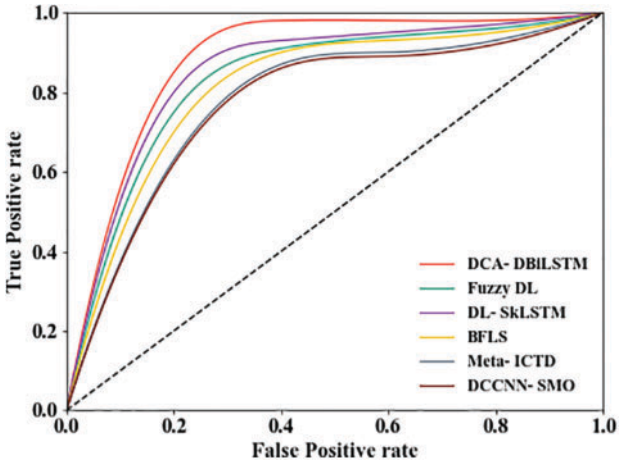

Figure 8: AUC-ROC analysis

Table 5 shows the computational time analysis of the DCA-DBiLSTM method with other existing methods. The study demonstrates that the DCA-DBiLSTM model is faster compared to other methods, achieving the lowest computational time of 2.31 s. This helps the model recognise threats early and mitigate hazards more effectively.

Table 5: Computational time analysis

Methods	Computational time (s)
DCA-DBiLSTM	2.31
Fuzzy DL	9.93
DL-SkLSTM	4.85
BFLS	11.84
Meta-ICTD	14.38
DCCNN-SMO	10.57

The Wilcoxon signed-rank test is employed to compare the effectiveness of different machine learning models, validating the accuracy of the research findings. As a non-parametric test, it facilitates robust inferences about performance differences across models. The last row of Table 6 shows the p-values for the Wilcoxon signed two-tailed rank test, which shows the significance level of statistical comparison between DCA-DBiLSTM and other models. The comparison of models presented in the study is considered statistically significant if the p-value is less than 0.05. From the table, it can be noted that the DCA-DBiLSTM model outperforms all the other models. The p-values for all pairwise comparisons are less than 0.05, which suggests that the differences in performance are not due to chance. Therefore, the statistical test confirms that, in this study, the DCA-DBiLSTM model is superior to the other existing models.

Table 6: Statistical analysis using the wilcoxon signed two-tailed rank test

No.	Pairwise model comparison	*p*–Value	Significance < 0.05
1	DCA-DBiLSTM vs. Fuzzy DL	0.0119	Yes
2	DCA-DBiLSTM vs. DP-NC	0.0021	Yes
3	DCA-DBiLSTM vs. BFLS	0.0070	Yes
4	DCA-DBiLSTM vs. Meta-ICTD	0.0044	Yes
5	DCA-DBiLSTM vs. DCCNN-SMO	0.0028	Yes

4.6 Ablation Study

The ablation study helps to understand the importance of each component in the overall performance of the DCA-DBiLSTM. This study is carried out to examine the contribution of each element present in the proposed model. The ablation study of the proposed model with the required modules is described in Table 7. In the ablation study, the components in the proposed model are removed one by one to find out how they impact DCA-DBiLSTM's performance. The table below illustrates that DCA-DBiLSTM did not perform well without preprocessing, feature selection, and feature extraction. Also, the simulation results demonstrate that without these components, the accuracy rate of the model was significantly reduced. However, the proposed model attains excellent accuracy with the implementation using all of these components. The ablation study indicates that each of these components is essential to achieve better recognition results.

Table 7: Ablation study of the DCA-DBiLSTM model

Ablation variant	Measures			
	Accuracy	Precision	Recall	F1 Score
Proposed	98.96	98.34	98.69	97.73
Without preprocessing	94.65	94.18	94.24	93.57
With preprocessing	95.89	95.37	95.52	94.76
With dual-channel attention mechanism	97.14	96.57	96.82	95.96
Without dual-channel attention mechanism	93.83	93.31	93.47	92.75

5 Conclusion

This paper proposed a novel approach for malware detection and risk mitigation using a Dual-Channel Attention Deep Bidirectional Long Short-Term Memory model. This model merges the potentiality of deep learning with attention mechanisms to improve detection abilities. The Bidirectional LSTM framework is employed, which processes the information in both directions (forward and backward) to help capture crucial dependencies from past and future states. This is essential for grasping the subtle behaviours of malware. The Dual Channel Attention mechanism used improves the model's capability to concentrate on the features that are most appropriate in the input data, which enhances the selection of features and reduces the possibility of false identifications. This integration of Bidirectional LSTM with DCA enables the model to learn sophisticated patterns that are related to both benign and malicious applications and hence is less sensitive to variability in different datasets. Analysing the detected and evaluated malware, risk management is invoked, where possible threats and vulnerabilities are systematically reduced through measures grounded on the severity and impact of the threats. To train the model, datasets such as the

CICAndMal2017 dataset and the Android Malware Detection datasets are utilized. The results of various experiments show that the model demonstrates superior performance and is effective in malware detection and risk mitigation, achieving greater accuracy, precision, recall, and F1 score values of 98.96%, 98.34%, 98.69%, and 97.73%, respectively. Since the model utilizes only two datasets, the future scope would focus on using multiple datasets considering all the characteristics to strengthen security mechanisms more effectively and provide an early response to mitigate threats platform independently. Future enhancements for the DCA-DBiLSTM framework could focus on improving scalability and adaptability to larger, more varied datasets by incorporating distributed training techniques, such as federated learning, to process data across multiple devices without compromising privacy.

Acknowledgement: The authors, therefore, acknowledge DSR's technical and financial support with thanks.

Funding Statement: This Project was funded by the Deanship of Scientific Research (DSR) at King Abdulaziz University, Jeddah, under grant No. (IPP: 421-611-2025).

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization, Madini O. Alassafi and Syed Hamid Hasan; methodology, Syed Hamid Hasan; software, Syed Hamid Hasan; validation, Madini O. Alassafi and Syed Hamid Hasan; formal analysis, Syed Hamid Hasan; investigation, Syed Hamid Hasan; resources, Madini O. Alassafi; data curation, Syed Hamid Hasan; writing—original draft preparation, Syed Hamid Hasan; writing—review and editing, Madini O. Alassafi and Syed Hamid Hasan; visualization, Syed Hamid Hasan; supervision, Madini O. Alassafi; project administration, Madini O. Alassafi; funding acquisition, Madini O. Alassafi. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The datasets used in this study are publicly available. The CICAndMal2017 dataset, provided by the Canadian Institute for Cybersecurity, can be accessed at https://www.unb.ca/cic/datasets/andmal2017.html (accessed on 01 January 2025). The Android Malware Detection dataset is available on Kaggle at https://www.kaggle.com/datasets/subhajournal/android-malware-detection (accessed on 01 January 2025).

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Sewak M, Sahay SK, Rathore H. Deep reinforcement learning in the advanced cybersecurity threat detection and protection. Inf Syst Front. 2023;25(2):589–611. doi:10.1007/s10796-022-10333-x.
2. Oyinloye TS, Arowolo MO, Prasad R. Enhancing cyber threat detection with an improved artificial neural network model. Data Sci Manag. 2025;8(1):107–15. doi:10.1016/j.dsm.2024.05.002.
3. Halbouni A, Gunawan TS, Habaebi MH, Halbouni M, Kartiwi M, Ahmad R. Machine learning and deep learning approaches for cybersecurity: a review. IEEE Access. 2022;10:19572–85. doi:10.1109/ACCESS.2022.3151248.
4. Taheri R, Shojafar M, Arabikhan F, Gegov A. Unveiling vulnerabilities in deep learning-based malware detection: differential privacy driven adversarial attacks. Comput Secur. 2024;146(2s):104035. doi:10.1016/j.cose.2024.104035.
5. Aslan Ö., Aktuğ SS, Ozkan-Okay M, Yilmaz AA, Akin E. A comprehensive review of cyber security vulnerabilities, threats, attacks, and solutions. Electronics. 2023;12(6):1333. doi:10.3390/electronics12061333.
6. Abdelkader S, Amissah J, Kinga S, Mugerwa G, Emmanuel E, Mansour DE, et al. Securing modern power systems: implementing comprehensive strategies to enhance resilience and reliability against cyber-attacks. Results Eng. 2024;23:102647. doi:10.1016/j.rineng.2024.102647.
7. Alazab M, Khurma RA, García-Arenas M, Jatana V, Baydoun A, Damaševičius R. Enhanced threat intelligence framework for advanced cybersecurity resilience. Egypt Inform J. 2024;27:100521. doi:10.1016/j.eij.2024.100521.
8. Hnamte V, Hussain J. Enhancing security in software-defined networks: an approach to efficient ARP spoofing attacks detection and mitigation. Telemat Inform Rep. 2024;14(13):100129. doi:10.1016/j.teler.2024.100129.

9. Srivastava A, Parmar V, Patel S, Chaturvedi A. Adaptive cyber defense: leveraging neuromorphic computing for advanced threat detection and response. In: 2023 International Conference on Sustainable Computing and Smart Systems (ICSCSS); 2023 Jun 14–16; Coimbatore, India. p. 1557–62. doi:10.1109/ICSCSS57650.2023.10169393.
10. Yazdinejad A, Kazemi M, Parizi RM, Dehghantanha A, Karimipour H. An ensemble deep learning model for cyber threat hunting in industrial internet of things. Digit Commun Netw. 2023;9(1):101–10. doi:10.1016/j.dcan.2022.09.008.
11. Yazdinejad A, Dehghantanha A, Parizi RM, Srivastava G, Karimipour H. Secure intelligent fuzzy blockchain framework: effective threat detection in IoT networks. Comput Ind. 2023;144:103801. doi:10.1016/j.compind.2022.10380.
12. Bhandari G, Lyth A, Shalaginov A, Grønli TM. Distributed deep neural-network-based middleware for cyber-attacks detection in smart IoT ecosystem: a novel framework and performance evaluation approach. Electronics. 2023;12(2):298. doi:10.3390/electronics12020298.
13. Baldini G. Detection of cybersecurity spoofing attacks in vehicular networks with recurrence quantification analysis. Comput Commun. 2022;191(1):486–99. doi:10.1016/j.comcom.2022.05.021.
14. Rajak A, Tripathi R. DL-SkLSTM approach for cyber security threats detection in 5G enabled IIoT. Int J Inf Technol. 2024;16(1):13–20. doi:10.1007/s41870-023-01651-7.
15. Jiang T, Shen G, Guo C, Cui Y, Xie B. BFLS: blockchain and federated learning for sharing threat detection models as cyber threat intelligence. Comput Netw. 2023;224(1):109604. doi:10.1016/j.comnet.2023.109604.
16. Dey AK, Gupta GP, Sahu SP. A metaheuristic-based ensemble feature selection framework for cyber threat detection in IoT-enabled networks. Decis Anal J. 2023;7:100206. doi:10.1016/j.dajour.2023.100206.
17. Vijayalakshmi P, Karthika D. Hybrid dual-channel convolution neural network (DCCNN) with spider monkey optimization (SMO) for cyber security threats detection in internet of things. Meas Sens. 2023;27(1):100783. doi:10.1016/j.measen.2023.100783.