Hybrid Deep Learning Network Intrusion Detection System Based on Convolutional Neural Network and Bidirectional Long Short-Term Memory

Anindra Ageng Jihado * and Abba Suganda Girsang

Computer Science Department, Bina Nusantara University, Jakarta 11480, Indonesia
Email: anindra.jihado@binus.ac.id (A.A.J.); agirsang@binus.edu (A.S.G.)
*Corresponding author

***Abstract*—Network security has become crucial in an era where information and data are valuable assets. An effective Network Intrusion Detection System (NIDS) is required to protect sensitive data and information from cyberattacks. Numerous studies have created NIDS using machine learning algorithms and network datasets that do not accurately reflect actual network data flows. Increasing hardware capabilities and the ability to process big data have made deep learning the preferred method for developing NIDS. This study develops a NIDS model using two deep learning algorithms: Convolutional Neural Network (CNN) and Bidirectional Long-Short Term Memory (BiLSTM). CNN extracts spatial features in the proposed model, while BiLSTM extracts temporal features. Two publicly available benchmark datasets, CICIDS2017 and UNSW-NB15, are used to evaluate the model. The proposed model surpasses the previous method in terms of accuracy, achieving 99.83% and 99.81% for binary and multiclass classification on the CICIDS2017 dataset. On the UNSW-NB15 dataset, the model achieves accuracies of 94.22% and 82.91% for binary and multiclass classification, respectively. Moreover, Principal Component Analysis (PCA) is also used for feature engineering to improve the speed of model training and reduce existing features to ten dimensions without significantly impacting the model's performance.**

***Keywords*—bidirectional long short-term memory, convolutional neural network, deep learning, network intrusion detection system, principal component analysis**

I. Introduction

Information technology, specifically the Internet, has become an integral component of daily life. Due to the widespread adoption of information technology in all sectors, numerous applications and data are interconnected on a network, making it susceptible to cyberattacks. To disrupt the confidentiality, integrity, and availability of the network's data, attackers may employ sniffer attacks, man-in-the-middle attacks, phishing, Distributed Denial-of-Service (DDoS), or the transmission of malware. In 2022, hacking and backdoors using malicious software continued to dominate security incidents. Furthermore, ransomware grew by nearly 13%, equivalent to the increase over the previous five years [1]. In the healthcare industry, implementing network security tools is imperative to protect healthcare data against unauthorized access and disclosure under the legal, ethical, and medical domains [2]. Hence, developing an effective network security tool to protect data and detect such threats is vital, as cyber-attack frequency and complexity rise yearly and are highly variable [3].

Network Intrusion Detection System (NIDS) is a security tool that analyzes network data flows to detect threats and prevent malicious requests [4]. Signature-based and anomaly-based NIDS are the two categories based on the detection method. Signature-based NIDS detects threats by comparing them to existing patterns or signatures in the database [5]. This type effectively identifies threats with low computation rates but has difficulty detecting new or zero-day attacks [6]. Anomaly-based NIDS, on the other hand, has the advantage of identifying new threats or unusual behavior but has a high false positive rate [7]. Thus, it is necessary to design a NIDS capable of detecting novel and established attacks with high accuracy and detection rates.

Researchers have extensively used machine learning techniques to develop anomaly-based NIDS [8]. This method consists of shallow learning and deep learning. Shallow learning has a short learning time and can learn essential features independently from other available features [9]. In contrast, deep learning requires a longer training time due to its complexity but has a more accurate representation of the data and can automatically extract features [10]. The use of deep learning has risen in recent years with the high volume of data processed and the development of hardware that can perform parallel computing and rapid processing [11]. Moreover, adaptive deep learning algorithms like CNN align well with the seamless adoption of 6G, attributed to its compatibility with a wide range of systems, services, device types, and

Manuscript received June 4, 2023; revised July 11, 2023; accepted October 7, 2023; published February 15, 2024.

data rates capable of reaching speeds up to 1 Tbps [12]. Deep learning's other advantages, such as its ability to learn from high-dimensional data and provide higher accuracy than shallow learning, add to its popularity in NIDS development studies.

Although deep learning has many advantages in developing NIDS, several developed deep learning NIDS models have low precision, a low detection rate, and a high detection error. Proper methodology and utilizing relevant and up-to-date datasets during model development are necessary to obtain accurate results when detecting threats and attacks [13]. Some models have been developed using obsolete datasets and do not accurately represent the current data flow on a network. Researchers still use more than two-decades-old datasets such as DARPA, MIT, and KDD CUP'99 [14]. The NIDS model will be unreliable when applied to the current, much more complex network data. The complexity of the dataset is another challenge in the NIDS model development. To reduce computational resources and processing time, the researchers experimented with various algorithms to select and reduce features in response to the large volume and quantity of data [15].

The main contributions of this paper are as follows:

- This paper proposes an effective hybrid NIDS model using a deep learning approach by employing CNN and Bidirectional Long-Short Term Memory (BiLSTM). The proposed model can automatically perform spatial and temporal feature extraction from network flow data.
- To tackle the complexity issues that arise from deep learning models, the Principal Component Analysis (PCA) technique is utilized for dimensionality reduction. We present an analysis conducted using several PCA components to assess the effectiveness and efficiency of the proposed model.
- To enhance the detection capabilities of state-of-the-art NIDS we address sample imbalance by relabeling dataset and utilizing more recent network datasets, namely CICIDS2017 and UNSW-NB15 as they have up-to-date common attacks. The results for binary and multiclass classification demonstrate that the proposed model outperforms previous existing models.

The structure of this paper is as follows: Section II provides an overview of the previous study that has been done. The proposed model's architecture and methodology are presented in Section III. Section IV describes the experiment and evaluation results of the model in comparison to other methods. Conclusions and suggestions for further study are described in Section V.

II. Related Works

With the development of NIDS, machine learning techniques such as Support Vector Machine (SVM) [16], Random Forest (RF) [17], eXtreme Gradient Boosting (XGBoost) [18], K-Means [19], and K-Nearest Neighbor (KNN) [20] are still utilized. However, with the growing amount of data and computing capabilities, deep learning is a more viable option, as it can process large amounts of data and represent raw data using multiple hidden layers and their deep structures [21].

In order to identify four different types of attacks on the CICIDS2017 dataset, Qazi *et al.* [22] developed the NIDS model using the deep learning method with the one-dimensional CNN architecture. The four types of attacks are Denial-of-Service (DoS) Hulk, DDoS, DoS Goldeneye, and Port Scan. The accuracy was 98.96% using five layers of CNN over 50 epochs. Al-Turaiki and Altwaijry [23] used CNN and Deep Feature Synthesis (DFS) to perform feature engineering. Additionally, PCA is used to reduce dimensions, while DFS is used to generate features. The classification is performed using a CNN with three to five convolutional layers for binary and multiclass classification.

The hybrid NIDS model has been developed using CNN, LSTM, and their combination by Halbouni *et al.* [24]. The CNN algorithm is used to extract spatial features, while the LSTM is used to extract temporal features. The evaluation was carried out using CICIDS2017, UNSW-NB15, and WSN-DS datasets. The result was that the CNN-LSTM model obtained the highest detection rate and accuracy. Han *et al.* [25] designed a NIDS model to address irregular time intervals between packets in flow and cutting or settling packet loads using 1D-CNN and Time and Length sensitive Long-Short Term Memory (TL-LSTM). The 1D-CNN model evaluation showed better results than 2D-CNN using CICIDS2017 and ISCX2012 datasets. The CNN-LSTM hybrid model was also made by Kim *et al.* [26]. The hybrid method is used to extract features from real-time HTTP traffic. Despite getting satisfactory results with 99% accuracy, existing systems must be re-validated due to existing false positive alarms. Khan *et al.* [27] developed an IDS with two stages: Spark ML for the anomaly detection module and Convolutional-LSTM for the misuse detection module. This combination results in a scalable and efficient IDS with an accuracy of 97.29%. An alternative hybrid deep learning model, BLoCNet, was developed by Bowen *et al.* [28]. It integrates a combination of CNN and two BiLSTM layers and achieved high detection accuracies of 98% and 76.34% on the CICIDS2017 and UNSW-NB15 datasets, respectively.

Another model with CNN-LSTM was developed to detect malicious request attacks on the Internet of Things (IoT) network by Yang *et al.* [29]. The system extracted semantic relationships and important features prior to classification using knowledge graphs. The CNN-BiLSTM-attention architecture can effectively capture long-term information, and the attention mechanism can highlight important features. Elsayed *et al.* [30] utilized the IoT Intrusion Dataset to develop an IDS explicitly designed for smart home environments. This IDS was created using the BiLSTM-CNN model. The proposed methodology outperforms existing state-of-the-art models and exhibits applicability to any smart home network gateway. Using more hidden layers, Hnamte and Hussain [31] integrate CNN with BiLSTM networks. The research focused on DoS attacks within the CICIDS2018 datasets and attacks on IoT systems using the Edge_IIoT dataset. Regrettably, due to the intricate nature of the model, further studies are necessary to facilitate its real-time implementation.

Sharafaldin *et al.* [32] relabeled the CICIDS2017 dataset's classes to address dataset imbalances. Classes with few records, similar characteristics, and similar behaviors are combined into a new minority class. The prevalence ratio of the malicious class increases when combined. Jiang *et al.* [33] used hybrid sampling methods such as One Side Selection (OSS) and Synthetic Minority Over-sampling Technique (SMOTE). The combination of OSS and SMOTE, and the BiLSTM classifier creates a balanced dataset that improves model performance and reduces learning time. The datasets used in this study are UNSW-NB15 and NSL-KDD. The Adaboost IDS model created by Yulianto *et al.* [34] also benefits from using SMOTE in addition to PCA and EFS methods. By merging these three techniques, recall rises from 84% to a perfect 100%, while the F1-Score improves from 77% to 90.01%.

A study on the reduction of features in NIDS development was conducted by Xiao *et al.* [35] using Auto Encoder (AE) and PCA. AE methods effectively eliminate redundancy in network traffic data, thereby improving detection and accuracy by CNN classifier. The study conducted by Abdulhammed *et al.* [36] also used PCA and AE in performing feature reduction. The features in the CICIDS 2017 dataset were successfully reduced from 81 to 10 features without reducing the detection performance with the RF classifier. Swarna *et al.* [37] performed feature engineering using PCA-GWO and Deep Neural Network (DNN). This model effectively detects attacks on Internet of Medical Thing (IoMT). Dahou *et al.* [38] utilized a CNN as a feature extractor in their model and employed the Reptile Search Algorithm to identify the most significant features. The RSA optimization algorithm outperforms the Particle Swarm Optimization (PSO), Firefly Algorithm (FFA), Grey Wolf Optimization (GWO), Whale Optimization Algorithm (WOA), Transient Search Optimization (TSO), Bat algorithm (BAT), and Multiverse Optimization algorithm (MVO).

To reduce false alerts in IDS, Andrew and Kathrine [39] used clustering and prioritizing methods. This method reduced false positive alerts to 67.50%, thus reducing the burden on network analysts in analyzing it.

Many researchers have moved beyond shallow learning to deep learning algorithms for NIDS development. Deep learning has become a hot topic in developing effective NIDS for detecting both normal and malicious network data flows. Another trend that is gaining prominence is the utilization of edge computing to support smart cities [40]. IDS researchers strive to design a system with minimal latency and limited resources. Numerous studies on the subject demonstrate that the structure of the NIDS model developed using CNN has been effective. Furthermore, researchers have experimented with hybrid structures combining two deep learning algorithms to improve detection performance. However, using a hybrid structure will increase model complexity and computational costs, both of which must be resolved if it is to be implemented practically. On the other hand, using outdated data can result in inaccurate evaluations. Although the evaluation results are positive, NIDS will be less effective at detecting recent attacks with variable patterns when implemented in the current environment.

The objective of this study is to create a hybrid deep learning model structure that surpasses previous models in terms of performance and efficiency. The hybrid model can capture spatial and temporal data characteristics to improve detection performance. PCA technique is utilized to make the model more efficient. In addition, this study employs more recent and representative real-world data to ensure future practical application.

III. Methodology

This study employed CNN and BiLSTM layers to develop a NIDS model. This combined approach enhances the model's ability to detect a wide range of network intrusions effectively.

CNN has an advantage over statistically stable and locally relevant data. The key benefit of CNN is its ability to recognize spatial features automatically without human intervention, avoid overfitting by reducing the number of parameters, and improve generalization. CNN also employs weight sharing to accelerate the model's training. Due to this benefit, the first layer is CNN, which extracts high-level features from massive network data streams via convolutional and max pooling layers. Convolutional layers utilize filters to extract the most critical features from network traffic, creating feature maps. Feature maps will then pass max pooling to save the most dominant features. Nevertheless, CNN faces challenges when capturing long-term information because it cannot analyze the connections between sequences of information. Consequently, the output is then forwarded to the BiLSTM layer for further analysis.

BiLSTM is capable of capturing long-distance dependence and information context. The information is extracted as a feature at the time level. BiLSTM is good at detecting continuous attacks [10]. Therefore, some network attacks, such as DoS, reconnaissance, and exploits, are more easily detected with BiLSTM. Features that are inherently tied to the temporal dimension find their representation within the BiLSTM. BiLSTM can capture the relationship between the number of host connections and the flag-specific characteristic of connection rejection in DoS attacks. This context must be considered in its entirety in order to make an accurate decision. Graves and Schmidhuber [41] showed that BiLSTM can considerably enhance classification accuracy. After all, it can capture the temporal dynamics of the system at the time of learning because it is performed in both forward and backward directions. By integrating LSTM, the model can effectively analyze the temporal dynamics of network traffic features.

Finally, the classification is done in fully connected layer using the extracted features. The framework model developed is seen in Fig. 1.

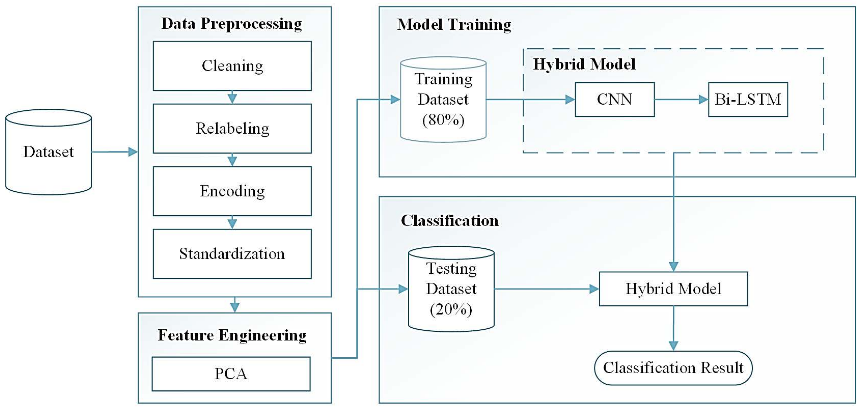


Fig. 1. The framework of NIDS.

A. *Dataset Preparation*

To obtain a model with excellent performance, choosing recent datasets and representing the network data streams is necessary. CICIDS2017 and UNSW-NB15 are the datasets used to construct this study's models. Both datasets are accessible to the public and are considered to represent actual attacks on a computer network.

1) CICIDS2017

The CICIDS2017 dataset was developed in response to the shortage of public datasets for designing NIDS. This dataset satisfies eleven of the framework's ideal criteria for evaluating datasets [42]. This dataset contains 78 features and 15 classes, of which one is benign, and 14 are attacks. DoS Hulk, Port Scan, DDoS, DoS GoldenEye, FTP Patator, SSH Patator, DoS Slow Loris, DoS Slow HTTP Test, Botnet, Web Attack: Brute Force, Web Attack: XSS, Infiltration, Web Attack: SQL Injection, and HeartBleed are the types of attacks that exist on this dataset.

This dataset contains several shortcomings, including the occurrence of class imbalances and the presence of both empty and redundant records [43]. To address these, it is necessary to preprocess the data by balancing the classes and cleaning the records.

2) UNSW-NB15

This dataset was created to represent normal and abnormal network data flows in the synthesis environment of the UNSW cyber security lab to be used as a benchmark in the development of NIDS [44, 45]. The UNSW-NB15 dataset consists of 49 features and nine attack classifications. Available attack types include Fuzzers, Analysis, Backdoor, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms. This dataset contains 2,540,044 records, which are stored in CSV format.

This study used a subset of the UNSW-NB15 dataset, specifically the UNSW_NB15_training-set.csv, and UNSW_NB15_testing-set.csv, to develop the model. There are a total of 175,341 records in this dataset. The drawback of this dataset is that the training set contains up to 42.24% duplication [46].

B. *Dataset Preprocessing*

1) Cleaning dataset

The CICIDS2017 data set is available to the public in CSV format and has been formatted for machine learning and deep learning techniques. The dataset is comprised of eight separate files that are merged into one CSV file. Cleaning was performed to delete empty and infinite values. We also eliminated redundant features, resulting in 77 features and 2,830,743 records. For the UNSW-NB15 dataset, we use 41 features, as there are unnecessary features such as row numbers and IDs.

2) Data relabeling

The relabeling process is carried out on a multiclass classification in the CICIDS2017 dataset. Relabeling made the dataset more balanced by changing the labels from 15 classes to 7 classes, as shown in Table I. Relabeling is needed to address class imbalances in the CICIDS2017 dataset. The relabeling increased the ratio of minority classes from 0.00039% to 0.001%.

TABLE I. CICIDS2017 CLASS RELABELING

No.	New Label	Old Label	#Records	Prevalence
1	Normal	BENIGN	2,273,097	80.300%
2	Botnet ARES	Bot	1,966	0.0695%
3	Brute Force	FTP-Patator SSH-Patator	13,835	0.4887%
4	DoS/DDoS	DDoS DoS Hulk DoS GoldenEye DoS slowloris DoS Slowhttptest Heartbleed	380,699	13.4487%
5	Infiltration	Infiltration	36	0.0013%
6	PortScan	PortScan	158,930	5.6144%
7	Web Attack	Web Attack-Brute Force Web Attack-XSS Web Attack-Sql Injection	2,180	0.0770%

3) Data encoding

In the dataset, not all features have numerical values. Some features, such as protocol, service, state, and labels, are not numerical. This phase aims to convert non-numerical values into numerical ones using One-Hot Encoder so it can be processed with a deep learning model. When one hot encoder is applied to the UNSW-NB15, the existing features are expanded, bringing the total number of features to 196.

4) *Data standardization*

The average distribution and standard deviation will affect the learning efficiency of the model. Standardizing data can avoid outliers, and CNN convergence can be reached more quickly. Data standardization is helpful at the feature reduction stage using PCA so that covariance can be easily compared with each pair of features.

Standardization transforms feature values so that the mean is 0 and the standard deviation is 1 using the Z-score shown in Eq. (1), where z is the normalization value, x is the starting value, μ is the population average, and σ is the population standard deviation.

$$z = \frac{x - \mu}{\sigma} \tag{1}$$

5) *Feature engineering*

In this phase, dimension reduction is performed using the well-known technique known as PCA. Features or variables will be transformed into a lower-dimensional space without losing valuable information, thereby reducing processing time, minimizing noise in data, and facilitating the visualization and comprehension of data. The number of features within the CICIDS2017 dataset will be reduced to 10, 30, and 50 principal components. For the UNSW-NB15 dataset, the existing features will be reduced to 10, 30, 50, and 100 principal components.

6) *Data splitting*

For the training and testing of the model, the dataset will be split 80:20. Furthermore, the training set will be split into training data and validation data in the same ratio. This ratio is used due to the large number of records in the dataset.

C. *Hybrid Deep Learning Model*

The design of the NIDS model combines the deep learning techniques of CNN and BiLSTM. The CNN layer can extract spatial features, whereas the BiLSTM layer can extract temporal features. Initially, inputs were processed with CNN to extract spatial features using a kernel or filter, resulting in feature maps. Feature maps are then forwarded to the subsequent layer, where batch normalization occurs, and then processed by max pooling to obtain the maximum value for each region. This procedure may be repeated one to three times before the output is sent to the BiLSTM layer for temporal feature extraction. The final step is a fully connected layer with three layers, where the last layer uses the activation function Sigmoid or Softmax. The structure of the CNN and BiLSTM hybrid models is depicted in Fig. 2.

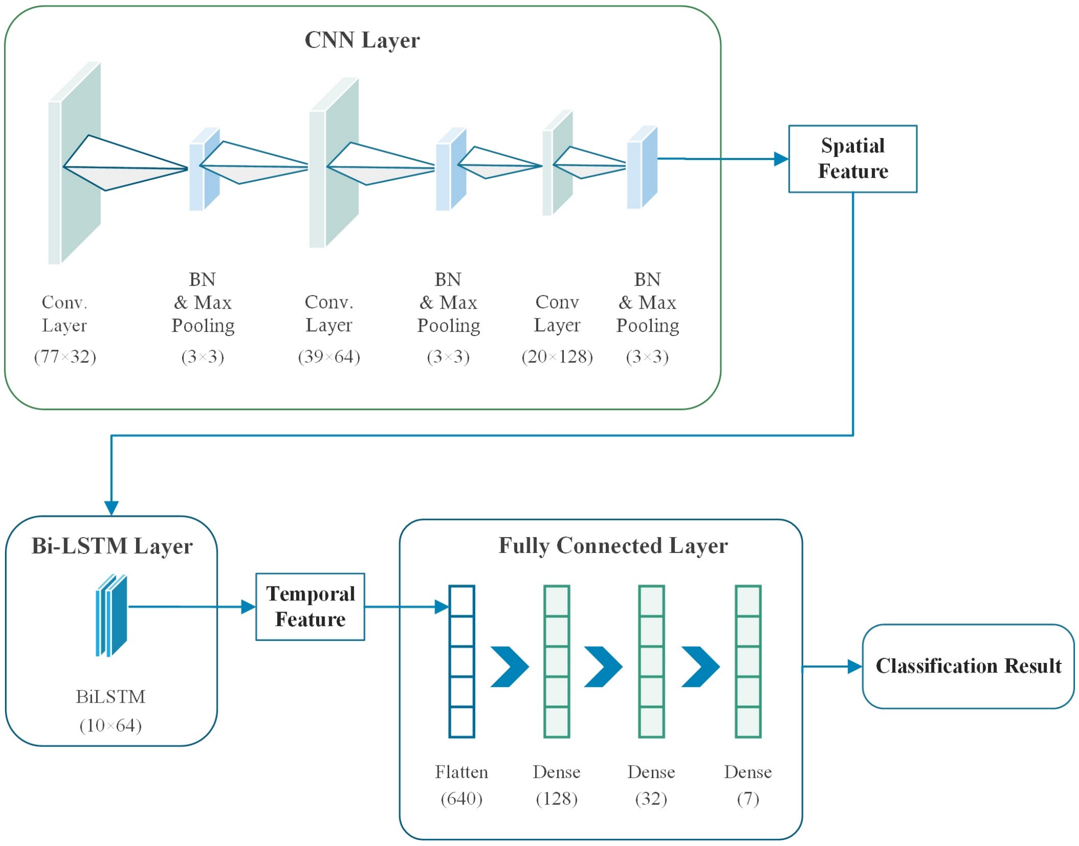


Fig. 2. CNN-BiLSTM architecture.

1) *Convolutional Neural Network (CNN)*

Two main layers are used on CNN: the convolutional and the pooling layers. On the convolutional layer, a kernel maps the input into an abstraction called a feature map or activation map, which is then sent to the next layer. The kernel maps the input by performing a dot operation on each element covered in the kernel and then summing it. Next, the kernel will shift horizontally and vertically until all the input is fully covered. The convolution process is shown in Fig. 3. The convolution process on the model plays a role in extracting spatial features so that CNN can study the hierarchical representation of the input automatically. The result of the conversion is then downsampled through the max pooling layer.

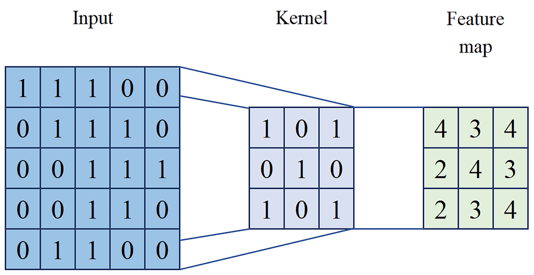


Fig. 3. Convolutional process on CNN.

The max pooling layer uses the largest value on the feature map it covers to reduce the matrix's dimension further. The reductions performed on max pooling had no effect on the weights [47]. Max pooling is also used to prevent model overfitting.

Mohammadpour *et al.* [48] have used one to three convolutional layers in the NIDS model. The proposed model would use one to three convolutional layers and maximum pooling to achieve the highest accuracy. The number of kernels on the three convolutional layers is 32, 64, and 128; kernel dimensions are 3×3 and use the same padding.

The activation function used is $ReLU\ (z_i) = max\ (0, z_i)$. The resulting feature map can be represented by Eq. (2), where h is the activation function, w_i is the weights, v_i is the data input, b is the bias, and $p \times q$ is the dimension of the input matrix. Before passing through the max pooling, the output of the convolutional layer will undergo the process of batch normalization.

$$Z = h\left(\sum_{i}^{p\times q} w_i v_i + b\right) \tag{2}$$

The CNN model is considered a black box model due to its high-dimensional feature space, which poses challenges for direct human interpretation of the acquired features. Because spatial features in the dataset used for this study are not stated explicitly, CNN will unaidedly extract and utilize spatial features from the dataset. Nevertheless, certain spatial feature considerations exist in the dataset, such as the Destination Port, Source Port, and Protocol.

2) *Batch normalization*

The learning process can be complex and slow due to the large number of layers on the network, which causes internal shifts in the covariate, i.e., the distribution of input from each layer changes as weights change in the previous layer. That is why normalization or regularization is carried out, which can mitigate covariate shifts and prevent overfitting from occurring.

Batch normalization is performed between the convolution layer and max pooling. Batch normalization is necessary to improve the performance of the training process. It applies a normalization operation that scales and shifts the inputs so that they have zero mean and unit variance. Batch normalization ensures that between existing layers, there is a similar range following the Gaussian distribution so that the optimization process runs more efficiently.

3) *Bidirectional long short-term memory*

Initially, RNN was state of the art in performing sequential data processing. RNN is widely used in speech recognition, natural language processing, time series analysis, and handwriting recognition. The advantage of RNN is that it can capture temporal dependencies in the input sequence to have memory of the previous input. However, RNN has limitations in capturing long-term dependencies. This occurrence is caused by the vanishing gradient problem, in which the gradient during backpropagation can diminish or become excessively large. As a solution to this problem, the LSTM architecture was developed.

LSTM allows long-term dependency modeling in sequential data using memory cells. Each memory cell has a gating mechanism to store or delete certain information from the previous time step. There are three main gateways for information processing: the forget gate, the input gate, and the output gate.

Forget gate determines which old information needs to be removed from the memory cell. This is done using the sigmoid function in Eq. (3), where t is timestep, W_f is weight, h_{t-1} is the previous hidden state, x_t is the input, and b_f is the bias.

$$f_t = \sigma\left(W_f.[h_{t-1}, x_t] + b_f\right) \tag{3}$$

Meanwhile, the input gate determines which new information is to be added to the memory cell. The information to be added is calculated with Eqs. (4) and (5). The result of an input gate is a point-by-point multiplication between i_t and $\check{C}_t$.

$$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i) \tag{4}$$

$$\check{C}_t = tanh(W_C.[h_{t-1}, x_t] + b_C) \tag{5}$$

After obtaining the information that needs to be removed or added, the cell state operation is performed by multiplying the previous cell state (C_{t-1}) with the value of forget gate (f_t) and then added with the value of input gate ($i_t \times \check{C}_t$), as shown in Eq. (6).

$$C_t = f_t \times C_{t-1} + i_t \times \check{C}_t \tag{6}$$

The output gate generates a value that contains information from the current and previous inputs to produce a new hidden state. The hidden state is obtained from the point-by-point multiplication between the output gate result (o_t) and the hyperbolic tangent function of cell state (C_t) as shown in Eqs. (7) and (8).

$$o_t = \sigma(W_0.[h_{t-1}, x_t] + b_0) \tag{7}$$

$$h_t = o_t \times tanh\ (C_t) \tag{8}$$

Unlike a standard LSTM, which processes a data point sequence only in one direction, the BiLSTM processes the sequence in two directions and combines its outputs. By using BiLSTM, information is obtained from the previous and subsequent input. Fig. 4 shows the BiLSTM architecture. The BiLSTM consists of two layers: a forward layer and a backward layer. The advantage of using these two layers is that the network gets more comprehensive information from the features.

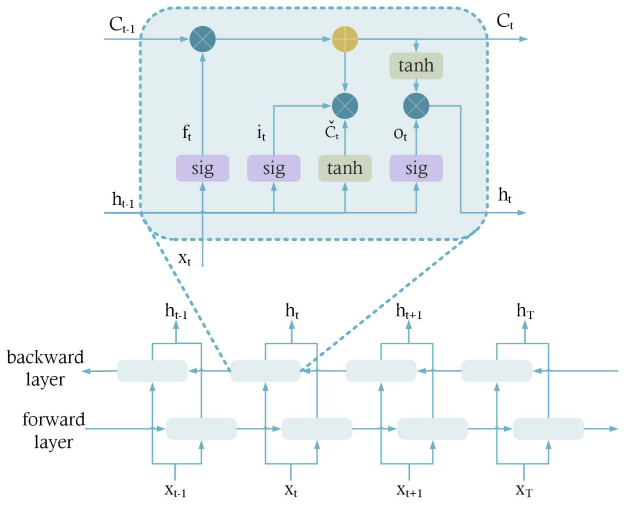


Fig. 4. BiLSTM structure.

The CICIDS2017 dataset does not contain any explicitly defined temporal features. However, temporal aspects can potentially be derived from the following set of features: Flow Duration, Flow Bytes/S, Flow Packet/S, Flow IAT Mean, Flow IAT Std, Flow IAT Max, Flow IAT Min, Fwd IAT Total, Fwd IAT Mean, Fwd IAT Std, Fwd IAT Max, Fwd IAT Min, Bwd IAT Total, Bwd IAT Mean, Bwd IAT Std, Bwd IAT Max, Bwd IAT Min, Fwd Packets/s, Bwd Packets/s, Fwd Avg Bulk Rate, Bwd Avg Bulk Rate, Active Mean, Active Std, Active Max, Active Min, Idle Mean, Idle Std, Idle Max, and Idle Min. These features provide valuable information concerning the timing, sequencing, or duration of events. BiLSTM will capture the necessary temporal features from the spatial features extracted from CNN. In the proposed model, 2×32 units are used, which are the dimensions of the output.

4) *Dropout*

Dropout is used to remove neurons from the network to prevent overfitting randomly. Random loss of neurons prevents models from relying on specific features and neurons. The model architecture will change as the active neurons will be different at each iteration, improving the generalization performance of the deep neural network model. In the proposed model, we added a dropout layer with a dropout rate 0.2 after the BiLSTM layer.

5) *Fully connected layer*

A fully connected layer or dense layer is a neural network in which each neuron on a layer connects with the neurons on the other layer. Weight and bias on fully connected layers are obtained from backpropagation and gradient descent. This layer is the final layer of the deep learning model. The flattening process converts the input dimension into a one-dimensional vector before entering a fully connected layer. We use three dense layers, with both the first and second dense having 128 and 32 neurons, and the last layer is used for classification by calculating the label probability of the input. In binary classification, the Sigmoid activation function is shown in Eq. (9), where *z* as an input will produce a value between 0 and 1. For multiclass classification, the Softmax activation function is shown in Eq. (10), where $(z)_i$ is the input vector, and *K* is the number of classes for classification.

$$\Phi(z) = \frac{1}{1+e^{-z}} \tag{9}$$

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{10}$$

D. *Evaluation*

As shown in Table II, the confusion matrix is used to evaluate the performance of the NIDS model. True Positive (*TP*) is an attack that has been classified correctly. False positive (*FP*) is the misclassification of benign data as an attack. True Negative (*TN*) is correctly classified benign data, whereas False Negative (*FN*) is a classification error in which an attack is misclassified as benign.

TABLE II. CONFUSION MATRIX

	Predicted Positive	**Predicted Negative**
Actual Positive	True Positive	False Negative
Actual Negative	False Positive	True Negative

We use accuracy, precision, recall, and the F1-Score to evaluate the performance of a model. Accuracy is the proportion of correctly classified data within a given dataset, as shown in Eq. (11). Precision is calculated with Eq. (12) by comparing the number of correctly classified malicious records to the total number of maliciously classified records. Recall is the proportion of malicious records correctly classified relative to the total number of malicious records and calculated with Eq. (13). The F1-Score, calculated using Eq. (14), combines precision and recall into a single value, offering a comprehensive evaluation of the model's performance.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{11}$$

$$Precision = \frac{TP}{TP+FP} \tag{12}$$

$$Recall = \frac{TP}{TP+FN} \tag{13}$$

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{14}$$

IV. RESULT AND ANALYSIS

We used a testbed Intel(R) Core (TM) i7-10700F CPU @ 2.90 GHz and 24 GB of RAM for the experiments. Implementation was done using Python 3.10 and Keras 2.11. Classification is carried out in two ways: binary and multiclass classification. The classes in each classification are shown in Table III. We use the Nadam optimizer with a learning rate of 0.001, 256 batch size, and 100 epochs during the training process.

TABLE III. DATASET CLASSIFICATION

Dataset	Classification			
	Binary		Multiclass	
	No. Class	Type	No. Class	Type
CICIDS 2017	2	Normal and attack	7	Normal, Botnet ARES, Brute Force, DoS/DDoS, Infiltration, Portscan, Web Attack
UNSW-NB15	2	Normal and attack	10	Normal, Generic, Exploits, Fuzzers, DoS, Reconnaissance, Analysis, Backdoor, Shellcode, Worms

A. Comparison Based on the Model's Structure with 1 to 3 CNN Layers

This phase focuses on developing the NIDS model's structure. To determine the optimal model structure, we compared the accuracy of CNN, BiLSTM, CNN-BiLSTM, and BiLSTM-CNN. Specifically, we add up to three layers to the CNN layer in each type to achieve the highest accuracy. Our experiment results are presented in Tables IV–VII.

TABLE IV. ACCURACY OF CICIDS2017 BINARY CLASSIFICATION BASED ON DIFFERENT MODEL STRUCTURES

No. CNN Layers	CNN	BiLSTM	CNN-BiLSTM	BiLSTM-CNN
0	*	99.67%	*	*
1	99.69%	*	99.83%	99.49%
2	99.76%	*	99.80%	99.65%
3	99.80%	*	99.75%	99.68%

TABLE V. ACCURACY OF CICIDS2017 MULTICLASS CLASSIFICATION BASED ON DIFFERENT MODEL STRUCTURES

No. CNN Layers	CNN	BiLSTM	CNN-BiLSTM	BiLSTM-CNN
0	*	99.61	*	*
1	99.72%	*	98.79%	99.68%
2	99.79%	*	99.80%	99.61%
3	99.80%	*	99.67%	99.67%

TABLE VI. ACCURACY OF UNSW-NB15 BINARY CLASSIFICATION BASED ON DIFFERENT MODEL STRUCTURES

No. CNN Layers	CNN	BiLSTM	CNN-BiLSTM	BiLSTM-CNN
0	*	93.98	*	*
1	93.83%	*	93.99%	94.04%
2	94.10%	*	94.15%	94.01%
3	93.97%	*	94.22%	94.06%

TABLE VII. ACCURACY OF UNSW-NB15 MULTICLASS CLASSIFICATION BASED ON DIFFERENT MODEL STRUCTURES

No. CNN Layers	CNN	BiLSTM	CNN-BiLSTM	BiLSTM-CNN
0	*	82.46%	*	*
1	82.29%	*	82.38%	82.66%
2	82.34%	*	82.58%	82.64%
3	82.48%	*	82.91%	82.61%

On the CICIDS2017 binary classification dataset, we obtained 99.83% accuracy using the CNN-BiLSTM with one CNN layer structure, followed by three CNN layers structure with 99.80% accuracy. Meanwhile, the BiLSTM-CNN model with a one-layer CNN structure obtained less satisfactory results with 99.49% accuracy.

For multiclass classification, the CNN-BiLSTM with two CNN layer's structure and the three CNN layer's structure achieved the highest accuracy of 99.80%. Structures that use the BiLSTM layer first to perform temporal feature extraction have a small accuracy ranging from 99.49% to 99.68%. This experiment on the CICIDS2017 dataset revealed that accuracy does not always increase as CNN layers are added.

Similar to CICIDS2017, the CNN-BiLSTM model structure achieves the highest accuracy on UNSW-NB15. The three-layer structure of CNN and BiLSTM can achieve the highest accuracy of 94.22% in binary classification and 82.91% in multiclass classification.

B. Evaluation Based on the Best Model's Structure

Once the optimal NIDS model structure was determined for each type, the next step in the experiment involved a detailed evaluation of the chosen structure, as presented in Table VIII. This task is crucial for evaluating each model's precision, recall, and F1-Score.

TABLE VIII. EVALUATION OF DIFFERENT MODEL STRUCTURES

Dataset	Classification	Model	Accuracy	Precision	Recall	F1-Score
CICIDS2017	Binary	CNN	99.80%	99.90%	99.86%	99.88%
		BiLSTM	99.67%	99.82%	99.77%	99.79%
		CNN-BiLSTM	99.83%	99.91%	99.87%	99.89%
		BiLSTM-CNN	99.68%	99.76%	99.83%	99.80%
	Multiclass	CNN	99.81%	99.80%	99.81%	99.80%
		BiLSTM	99.61%	99.60%	99.61%	99.58%
		CNN-BiLSTM	99.81%	99.80%	99.81%	99.80%
		BiLSTM-CNN	99.67%	99.66%	99.67%	99.63%
UNSW-NB15	Binary	CNN	93.83%	95.52%	94.79%	95.15%
		BiLSTM	93.98%	95.99%	94.52%	95.25%
		CNN-BiLSTM	94.22%	96.15%	94.76%	95.45%
		BiLSTM-CNN	94.06%	96.04%	94.60%	95.32%
	Multiclass	CNN	82.48%	82.40%	82.48%	80.04%
		BiLSTM	82.46%	81.80%	82.46%	80.76%
		CNN-BiLSTM	82.91%	82.38%	82.91%	80.94%
		BiLSTM-CNN	82.66%	82.91%	82.66%	79.99%

Fig. 5 is the result of the binary classification of CICIDS2017. In previous experiments, the CNN-BiLSTM model structure with the highest accuracy achieved precision, recall, and an F1-Score of 99.81%, 99.87%, and 99.89%, respectively. This metric is the highest among all tested models. The CNN-BiLSTM model also performed well in multiclass classification, scoring 99.80% accuracy, 99.81% recall, and 99.80% F1-Score, respectively. This value is comparable to the CNN model with three layers depicted in Fig. 6.

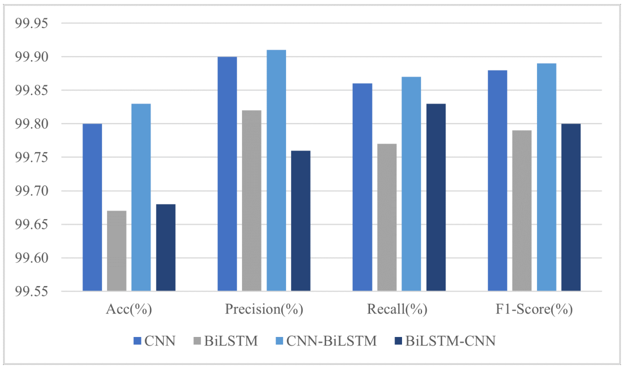


Fig. 5. CICIDS2017 binary classification evaluation result.

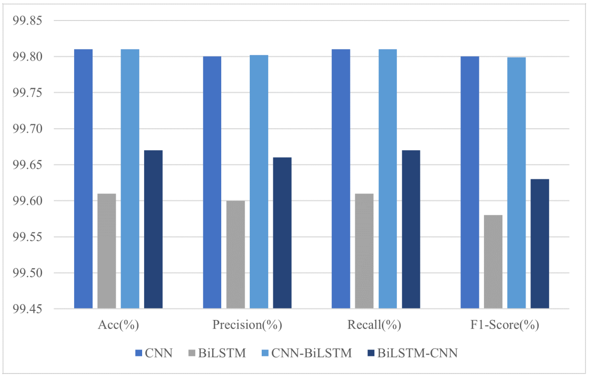


Fig. 6. CICIDS2017 multiclass classification evaluation result.

As shown in Fig. 7, CNN-BiLSTM obtained the highest precision, recall, and F1-Score scores on the UNSW-NB15 dataset for binary classification, with 96.15%, 94.7%, and 95.45%, respectively. BiLSTM-CNN has the highest precision scores for multiclass classification, at 82.91%, compared to CNN-BILSTM, which has only 82.38% accuracy. Fig. 8 shows that CNN-BiLSTM has the highest accuracy, recall, and F1-Score, with 82.91%, 82.91%, and 80.94%, respectively.

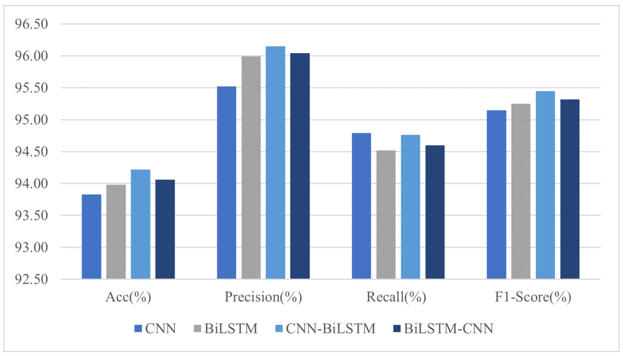


Fig. 7. UNSW-NB15 binary classification evaluation result.

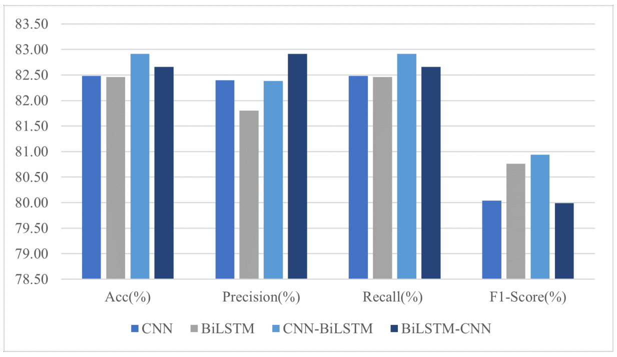


Fig. 8. UNSW-NB15 multiclass classification evaluation result.

Figs. 9 and 10 display the convergence graphs on the CICIDS2017 dataset. Fluctuations still occur before epochs 34 and 36 for binary and multiclass classification. On the other hand, the CNN-BiLSTM model for UNSW-NB15 demonstrates a noticeable advantage in terms of convergence speed for binary and multiclass classification tasks than the other models, as observed in Figs. 11 and 12, respectively. Specifically, this model begins to converge around epoch 45, whereas alternative models exhibit slower convergence beyond epoch 60.

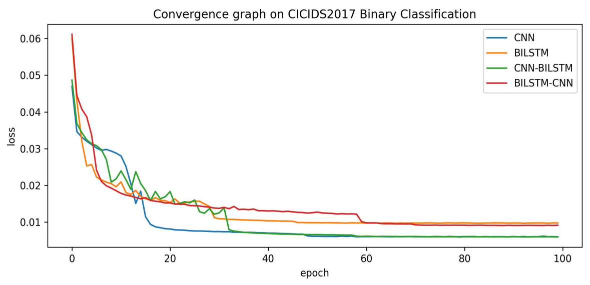


Fig. 9. Convergence graph for CICIDS2017 binary classification.

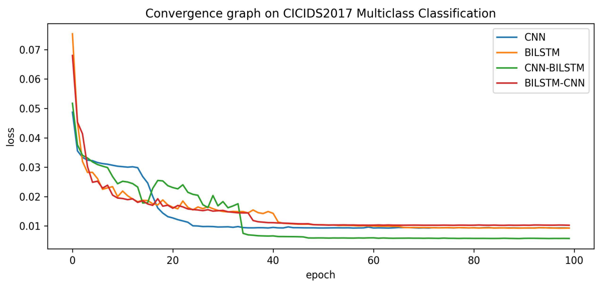


Fig. 10. Convergence graph for CICIDS2017 multiclass classification.

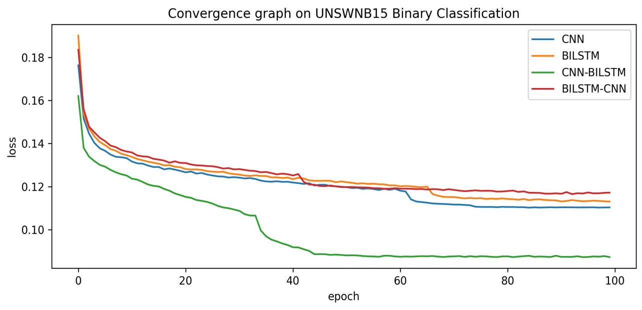


Fig. 11. Convergence graph for UNSW-NB15 binary classification.

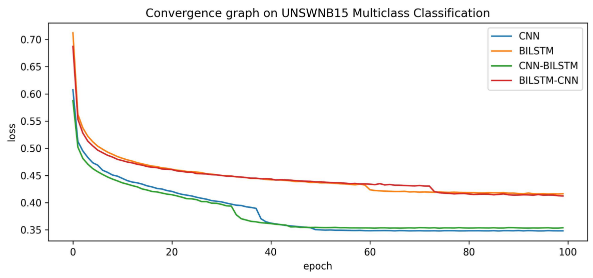


Fig. 12. Convergence graph for UNSW-NB.

Table IX shows the F1-Score for each class on the CICIDS2017. The F1-Score reflects a balance between recall and accuracy. Apart from DoS/DDoS, models with the CNN-BiLSTM structure achieved the highest F1-Scores in all classes. At 63.33%, the ARES botnet received the lowest F1-Score on this model. The model where BiLSTM is positioned as the initial layer exhibits an F1-Score of 19% on Web Attack, considerably inferior to the model where CNN is placed as the initial layer. Due to the limited number of classes in the dataset, the Infiltration class attains a relatively low F1-Score.

The F1-Score on the CNN-BiLSTM model on UNSW-NB15 outperforms other models in five classes: Normal, Analysis, Fuzzers, Shellcode, and Worms. The BiLSTM-CNN model has the highest F1-Score for the Backdoor, Reconnaissance, Exploits, and Generic classes, while the BiLSTM model has the highest F1-Score for the DoS class. Detecting the Analysis class proved to be the most challenging task, given the F1-Scores falling within the range of 9 to 13%. The complete F1-Score for UNSW-NB15 classes is shown in Table X.

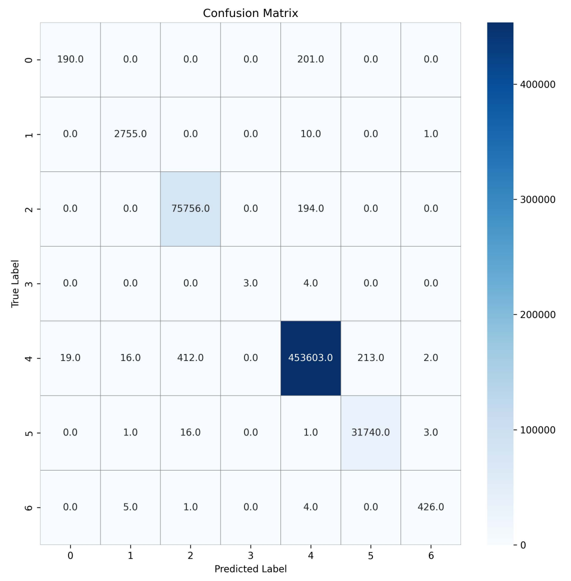
TABLE IX. F1-SCORE FOR EACH CLASS OF CICIDS2017

Class	CNN	BiLSTM	CNN-BiLSTM	BiLSTM-CNN
Botnet ARES	61.13%	60.56%	63.33%	59.75%
Brute Force	99.16%	98.87%	99.40%	98.96%
DoS/DDoS	99.61%	99.16%	99.59%	99.38%
Infiltration	44.44%	40.00%	60.00%	25.00%
Normal	99.88%	99.76%	99.88%	99.81%
Portscan	99.63%	99.59%	99.63%	99.64%
Web Attack	96.24%	19.18%	98.16%	19.83%

TABLE X. F1-SCORE FOR EACH CLASS OF UNSW-NB15

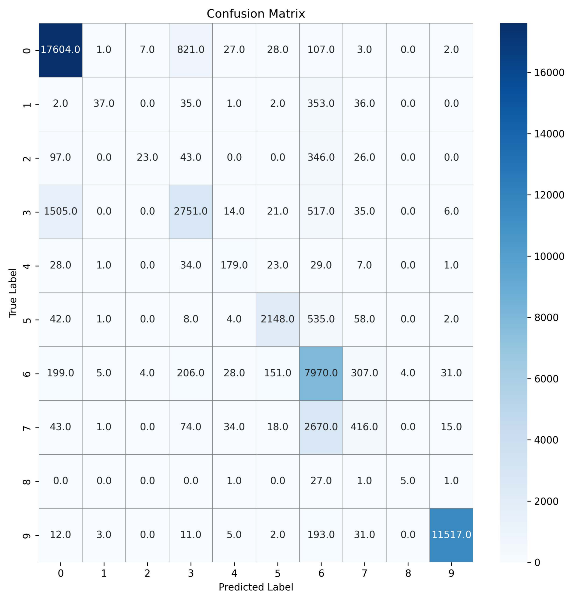
Class	CNN	BiLSTM	CNN-BiLSTM	BiLSTM-CNN
Normal	92.11%	92.01%	92.63%	92.10%
Analysis	11.36%	11.17%	13.38%	9.89%
Fuzzers	63.10%	61.29%	63.67%	60.80%
Shellcode	58.10%	57.28%	58.11%	55.96%
Backdoor	13.41%	14.50%	14.31%	14.94%
Reconnaissance	80.45%	81.72%	82.47%	82.58%
Exploits	73.26%	73.57%	73.50%	73.73%
DoS	11.32%	23.38%	18.61%	10.27%
Worms	5.26%	10.81%	14.29%	10.26%
Generic	98.62%	98.78%	98.64%	98.82%

Fig. 13 shows the confusion matrix for CICIDS2017, which indicates the number of records classified correctly or incorrectly in each class. The confusion matrix for UNSW-NB15 can be seen in Fig. 14. The Exploit class had the highest detection errors, with 2,692 records incorrectly classified as DoS.



True \ Predicted	0	1	2	3	4	5	6
0	190.0	0.0	0.0	0.0	201.0	0.0	0.0
1	0.0	2755.0	0.0	0.0	10.0	0.0	1.0
2	0.0	0.0	75756.0	0.0	194.0	0.0	0.0
3	0.0	0.0	0.0	3.0	4.0	0.0	0.0
4	19.0	16.0	412.0	0.0	453603.0	213.0	2.0
5	0.0	1.0	16.0	0.0	1.0	31740.0	3.0
6	0.0	5.0	1.0	0.0	4.0	0.0	426.0

Fig. 13. Confusion matrix for CICIDS2017 CNN-BiLSTM model.



True \ Predicted	0	1	2	3	4	5	6	7	8	9
0	17604.0	1.0	7.0	821.0	27.0	28.0	107.0	3.0	0.0	2.0
1	2.0	37.0	0.0	35.0	1.0	2.0	353.0	36.0	0.0	0.0
2	97.0	0.0	23.0	43.0	0.0	0.0	346.0	26.0	0.0	0.0
3	1505.0	0.0	0.0	2751.0	14.0	21.0	517.0	35.0	0.0	6.0
4	28.0	1.0	0.0	34.0	179.0	23.0	29.0	7.0	0.0	1.0
5	42.0	1.0	0.0	8.0	4.0	2148.0	535.0	58.0	0.0	2.0
6	199.0	5.0	4.0	206.0	28.0	151.0	7970.0	307.0	4.0	31.0
7	43.0	1.0	0.0	74.0	34.0	18.0	2670.0	416.0	0.0	15.0
8	0.0	0.0	0.0	0.0	1.0	0.0	27.0	1.0	5.0	1.0
9	12.0	3.0	0.0	11.0	5.0	2.0	193.0	31.0	0.0	11517.0

Fig. 14. Confusion Matrix for UNSW-NB15 CNN-BiLSTM model.

C. *CNN-BiLSTM Model Evaluation Based on PCA Components*

At this point, we decided to assess the efficiency of the PCA dimensionality reduction technique using the CNN-BiLSTM model. CICIDS2017 will use 10, 30, and 50 PCA components, while UNSW-NB15 will use 10, 30, 50, and 100 PCA components. The complete results are presented in Tables XI–XIV. The training time chart for each dataset based on the number of PCA components is shown in Figs. 15 and 16.

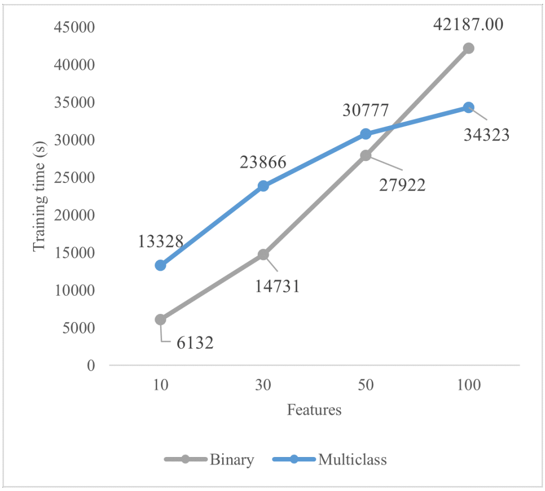


Fig. 15. Training time on CICIDS2017 based on the number of features.

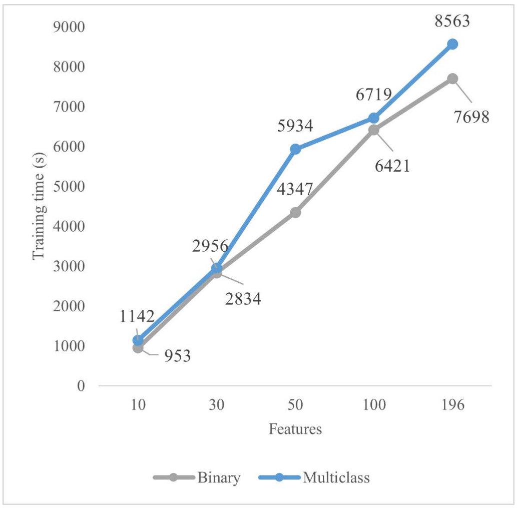


Fig. 16. Training time on UNSW-NB15 based on the number of features.

Using 50 PCA components, CNN-BiLSTM models with CICIDS2017 binary and multiclass classification achieve the highest accuracy, precision, recall, and F1-Score. Furthermore, using only 10 PCA components instead of 50 PCA components reduced training time by 80%. By using only 10 PCA components, the model's performance experienced a slight decrease in accuracy by approximately 0.5% to 1% while benefiting from reduced training time. This result is still acceptable because the model's performance has remained relatively high while benefiting from reduced training time.

For UNSW-NB15, given that the number of records is lower than in CICIDS2017, the difference in training time is not particularly noteworthy. Using 100 PCA components for both binary and multiclass classification produces superior performance in comparison to 10, 30, and 50 components. However, compared to using all 196 features of UNSW-NB15, the accuracy decreased to 92.98% and 81.04%, respectively. The best precision for binary classification is achieved with 50 PCA components with 94.69%, which is better than the model that uses 100 PCA components.

D. Benchmark Evaluation from Previous Studies

Tables XI–XVI compare our most effective CNN-BiLSTM model with previous studies. The proposed model outperformed previous studies, achieving an accuracy of 99.83% for the CICIDS2017 binary classification and 94.22% for the UNSW-NB15 binary classification. Regarding multiclass classification accuracy, CICIDS2017 achieved a remarkable accuracy of 99.81%, whereas UNSW-NB15 achieved an accuracy of 82.91%. This result suggests that combining the CNN layer and BiLSTM with our methodology can improve the performance of the classification model.

TABLE XI. DIMENSIONALITY REDUCTION ON CICIDS2017 BINARY CLASSIFICATION

PCA	Acc.	Prec.	Recall	F1-Score	Training Time (s)
10	99.32%	99.69%	99.46%	99.58%	6,132
30	98.90%	99.52%	99.11%	99.31%	14,731
50	99.80%	99.88%	99.87%	99.88%	27,922
Original	99.83%	99.91%	99.87%	99.89%	42,187

TABLE XII. DIMENSIONALITY REDUCTION ON CICIDS2017 MULTICLASS CLASSIFICATION

PCA	Acc.	Prec.	Recall	F1-Score	Training Time (s)
10	98.27%	98.32%	98.27%	98.26%	13,328
30	98.85%	98.88%	98.85%	98.82%	23,866
50	99.76	99.76	99.76	99.75%	30,777
Original	99.81	99.80	99.81	99.80%	34,323

TABLE XIII. DIMENSIONALITY REDUCTION ON UNSW-NB15 BINARY CLASSIFICATION

PCA	Acc.	Prec.	Recall	F1-Score	Training Time (s)
10	92.25%	94.15%	93.69%	93.92%	953
30	92.50%	94.03%	94.25%	94.14%	2,834
50	92.95%	94.69%	94.25%	94.47%	4,347
100	92.98%	94.61%	94.40%	94.50%	6,421
Original	94.06%	96.15%	94.76%	95.45%	7,698

TABLE XIV. DIMENSIONALITY REDUCTION ON UNSW-NB15 MULTICLASS CLASSIFICATION

PCA	Acc.	Prec.	Recall	F1-Score	Training Time (s)
10	80.17%	79.43%	80.17%	77.39%	1,142
30	80.54%	80.13%	80.54%	77.89%	2,956
50	80.47%	79.93%	80.47%	78.05%	5,934
100	81.04%	80.25%	81.04%	78.97%	6,719
Original	82.91%	82.38%	82.91%	80.94%	8,563

TABLE XV. PERFORMANCE COMPARISON ON PREVIOUS STUDIES FOR CICIDS2017

Classifier	Binary classification (%)				Multiclass classification (%)			
	Acc.	Prec.	Recall	F1-Score	Acc	Prec.	Recall	F1-Score
CNN-LSTM [24]	99.64%	99.56%	99.70%	99.30%	99.52%	99.42%	99.64%	99.22%
CNN-LSTM [25]	*	*	*	*	93.00%	86.47%	76.83%	81.36%
1D-CNN [48]	*	*	*	*	98.08%	92.48%	96.51%	94.45%
LSTM + 1D-CNN [48]	*	*	*	*	98.02%	90.38%	98.81%	94.41%
Proposed model	99.83%	99.91%	99.87%	99.89%	99.81%	99.80%	99.81%	99.80%

TABLE XVI. PERFORMANCE COMPARISON ON PREVIOUS STUDIES FOR UNSW-NB15

Classifier	Binary classification (%)				Multiclass classification (%)			
	Acc.	Prec.	Recall	F1-Score	Acc	Prec.	Recall	F1-Score
MCNN-DFS [23]	*	*	*	*	80.51%	81.00%	81.00%	81.00%
CNN-LSTM [24]	93.78%	94.69%	94.53%	93.88%	82.20%	82.12%	82.33%	80.43%
CNN-BiLSTM [33]	*	*	*	*	75.56%	79.33%	75.64%	77.44%
Proposed model	94.22%	96.15%	94.76%	95.45%	82.91%	82.38%	82.91%	80.94%

The performance of some models that use PCA to reduce features exceeds that of the model in previous studies. In the CICIDS2017 binary classification, a model using 50 PCA components obtained a 99.80% accuracy, which was still better than the CNN-LSTM model in other studies. An accuracy of 99.76% is also achieved in multiclass classification, surpassing the performance of CNN-LSTM and CNN-BiLSTM models in previous studies. On the multiclass classification, UNSW-NB15 models with 100 PCA components have not been able to beat the accuracy of CNN-LSTM in the previous study of 82.12%. Nevertheless, the proposed model, which utilizes the complete set of features, attained the highest level of performance, with an accuracy of 82.38% and a recall of 82.91%. Furthermore, the MCNN-DFS model achieved the highest F1-Score of 81%, close to the proposed model's score of 80.94%.

V. CONCLUSION

This study developed a network intrusion detection system using the deep learning algorithms CNN and BiLSTM. The CNN algorithm was used to extract spatial features, while BiLSTM was employed for temporal feature extraction. Initially, a comparison was made between the structures of CNN, BiLSTM, CNN-BiLSTM, and BiLSTM-CNN models, incorporating up to three CNN layers to assess their performance.

The evaluation of the models was carried out using two publicly available datasets: CICIDS2017 and UNSW-NB15. Both binary classification and multiclass classification were performed on these datasets. The hybrid CNN-BiLSTM model demonstrated superior performance on the CICIDS2017 dataset, achieving 99.83% and 99.81% accuracy in binary and multiclass classification, respectively. For the UNSW-NB15 dataset, the proposed model achieved 94.22% and 82.91% accuracy in binary and multiclass classification, respectively.

Combining the CNN and BiLSTM algorithms in the CNN-BiLSTM model resulted in a complex architecture. The application of the PCA technique was employed to accelerate the classification process. Although there was a slight decline in detection performance, the training time decreased significantly, up to seven times faster, when using 10 PCA components.

In future work, we will focus on augmenting the model's performance by addressing class imbalance within the dataset. Achieving a balanced dataset will require more than a mere process of relabeling. SMOTE will be applied for over-sampling the minority classes, and under-sampling will be implemented for the majority classes using clustering algorithms like GMM or the BIRCH. To ensure practical applicability, we intend to integrate our anomaly-based NIDS model with Snort, a signature-based NIDS, in future research. This amalgamation intends to make the most of the advantages of both systems, identifying new threats or unusual behavior while minimizing false positives.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

Anindra Ageng Jihado conducted the research and wrote the paper. Abba Suganda Girsang supervised the research and the paper. All authors had approved the final version.

REFERENCES

[1] S. Widup, A. Pinto, D. Hylender, G. Bassett, and P. Langlois. Data Breach Investigations Report. [Online]. Available: https://www.wired.com/images_blogs/threatlevel/2011/04/Verizon-2011-DBIR_04-13-11.pdf

[2] J. Andrew, R. J. Eunice, and J. Karthikeyan, "An anonymization-based privacy-preserving data collection protocol for digital health data," *Front Public Health*, vol. 11, Mar. 2023.

[3] O. Aslan and R. Samet, "A comprehensive review on malware detection approaches," *IEEE Access*, vol. 8, pp. 6249–6271, 2020.

[4] Z. Yang *et al*., "A systematic literature review of methods and datasets for anomaly-based network intrusion detection," *Comput. Secur*., vol. 116, 102675, May 2022.

[5] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: Techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, 20, Dec. 2019.

[6] M. Masdari and H. Khezri, "A survey and taxonomy of the fuzzy signature-based intrusion detection systems," *Appl. Soft. Comput*., vol. 92, 106301, Jul. 2020.

[7] S. Dwivedi, M. Vardhan, S. Tripathi, and A. K. Shukla, "Implementation of adaptive scheme in evolutionary technique for

anomaly-based intrusion detection," *Evol. Intell.*, vol. 13, no. 1, pp. 103–117, Mar. 2020.

[8] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 1, Jan. 2021.

[9] D. E. Kim and M. Gofman, "Comparison of shallow and deep neural networks for network intrusion detection," in *Proc. 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, 2018, pp. 204–208.

[10] Y. Zhong *et al*., "HELAD: A novel network anomaly detection model based on heterogeneous ensemble learning," *Computer Networks*, vol. 169, 107049, Mar. 2020.

[11] M. Pandey *et al*., "The transformational role of GPU computing and deep learning in drug discovery," *Nat. Mach. Intell.*, vol. 4, no. 3, pp. 211–221, 2022.

[12] V. K. Quy, A. Chehri, N. M. Quy, N. D. Han, and N. T. Ban, "Innovative trends in the 6G Era: A comprehensive survey of architecture, applications, technologies, and challenges," *IEEE Access*, vol. 11, pp. 39824–39844, 2023.

[13] I. Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. 4th International Conference on Information Systems Security and Privacy, SCITEPRESS - Science and Technology Publications*, 2018, pp. 108–116.

[14] Z. Inayat, A. Gani, N. B. Anuar, M. K. Khan, and S. Anwar, "Intrusion response systems: Foundations, design, and challenges," *Journal of Network and Computer Applications*, vol. 62, pp. 53–74, Feb. 2016.

[15] K. Kurniabudi, D. Stiawan, D. Darmawijoyo, M. Y. B. Idris, B. Kerim, and R. Budiarto, "Important features of CICIDS-2017 dataset for anomaly detection in high dimension and imbalanced class dataset," *Indonesian Journal of Electrical Engineering and Informatics (IJEEI)*, vol. 9, no. 2, May 2021.

[16] A. Khraisat, I. Gondal, P. Vamplew, J. Kamruzzaman, and A. Alazab, "Hybrid intrusion detection system based on the stacking ensemble of C5 decision tree classifier and one class support vector machine," *Electronics (Basel)*, vol. 9, no. 1, 173, Jan. 2020.

[17] S. Waskle, L. Parashar, and U. Singh, "Intrusion detection system using PCA with random forest approach," in *Proc 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*, 2020, pp. 803–808.

[18] S. Dhaliwal, A. A. Nahid, and R. Abbas, "Effective Intrusion detection system using XGBoost," *Information*, vol. 9, no. 7, 149, Jun. 2018.

[19] H. Yao, D. Fu, P. Zhang, M. Li, and Y. Liu, "MSML: A novel multilevel semi-supervised machine learning framework for intrusion detection system," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1949–1959, Apr. 2019.

[20] G. Liu, H. Zhao, F. Fan, G. Liu, Q. Xu, and S. Nazir, "An enhanced intrusion detection model based on improved KNN in WSNS," *Sensors*, vol. 22, no. 4, 1407, Feb. 2022.

[21] H. Liu and B. Lang, "Machine learning and deep learning methods for intrusion detection systems: A survey," *Applied Sciences*, vol. 9, no. 20, 4396, Oct. 2019.

[22] E. U. H. Qazi, A. Almorjan, and T. Zia, "A One-dimensional Convolutional Neural Network (1D-CNN) based deep learning system for network intrusion detection," *Applied Sciences*, vol. 12, no. 16, 7986, Aug. 2022.

[23] I. Al-Turaiki and N. Altwaijry, "A convolutional neural network for improved anomaly-based network intrusion detection," *Big Data*, vol. 9, no. 3, pp. 233–252, Jun. 2021.

[24] A. Halbouni, T. S. Gunawan, M. H. Habaebi, M. Halbouni, M. Kartiwi, and R. Ahmad, "CNN-LSTM: Hybrid deep neural network for network intrusion detection system," *IEEE Access*, vol. 10, pp. 99837–99849, 2022.

[25] X. Han *et al*., "STIDM: A spatial and temporal aware intrusion detection model," in *Proc. 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, 2020, pp. 370–377.

[26] A. Kim, M. Park, and D. H. Lee, "AI-IDS: Application of deep learning to real-time web intrusion detection," *IEEE Access*, vol. 8, pp. 70245–70261, 2020.

[27] M. Khan, Md. Karim, and Y. Kim, "A scalable and hybrid intrusion detection system based on the convolutional-LSTM network," *Symmetry (Basel)*, vol. 11, no. 4, 583, Apr. 2019.

[28] B. Bowen, A. Chennamaneni, A. Goulart, and D. Lin, "BLoCNet: A hybrid, dataset-independent intrusion detection system using deep learning," *Int. J. Inf Secur*., vol. 22, no. 4, pp. 893–917, Aug. 2023.

[29] X. Yang, G. Peng, D. Zhang, and Y. Lv, "An enhanced intrusion detection system for IoT networks based on deep learning and knowledge graph," *Security and Communication Networks*, pp. 1–21, Apr. 2022.

[30] N. Elsayed, Z. S. Zaghloul, S. W. Azumah, and C. Li, "Intrusion detection system in smart home network using bidirectional LSTM and convolutional neural networks hybrid model," in *Proc. 2021 IEEE International Midwest Symposium on Circuits and Systems (MWSCAS)*, 2021, vol. 13.

[31] V. Hnamte and J. Hussain, "DCNNBiLSTM: An efficient hybrid deep learning-based intrusion detection system," *Telematics and Informatics Reports*, vol. 10, 100053, Jun. 2023.

[32] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "A detailed analysis of the CICIDS2017 data set," *Information Systems Security and Privacy*, pp. 172–188, 2019.

[33] K. Jiang, W. Wang, A. Wang, and H. Wu, "Network intrusion detection combined hybrid sampling with deep hierarchical network," *IEEE Access*, vol. 8, pp. 32464–32476, 2020.

[34] A. Yulianto, P. Sukarno, and N. A. Suwastika, "Improving AdaBoost-based Intrusion Detection System (IDS) performance on CIC IDS 2017 dataset," *J. Phys. Conf. Ser*., vol. 1192, 012018, Mar. 2019.

[35] Y. Xiao, C. Xing, T. Zhang, and Z. Zhao, "An intrusion detection model based on feature reduction and convolutional neural networks," *IEEE Access*, vol. 7, pp. 42210–42219, 2019.

[36] R. Abdulhammed, H. Musafer, A. Alessa, M. Faezipour, and A. Abuzneid, "Features dimensionality reduction approaches for machine learning based network intrusion detection," *Electronics (Basel)*, vol. 8, no. 3, 322, Mar. 2019.

[37] P. R. M. Swarna *et al*., "An effective feature engineering for DNN using hybrid PCA-GWO for intrusion detection in IoMT architecture," *Comput. Commun*., vol. 160, pp. 139–149, Jul. 2020.

[38] A. Dahou *et al*., "Intrusion detection system for IoT based on deep learning and modified reptile search algorithm," *Comput. Intell. Neurosci*., vol. 2022, pp. 1–15, Jun. 2022.

[39] A. J. and G. J. W. Kathrine, "An intrusion detection system using correlation, prioritization and clustering techniques to mitigate false alerts," *Advances in Big Data and Cloud Computing*, pp. 257–268, 2018.

[40] Q. V. Khanh, V. H. Nguyen, Q. N. Minh, A. D. Van, N. L. Anh, and A. Chehri, "An efficient edge computing management mechanism for sustainable smart cities," *Sustainable Computing: Informatics and Systems*, vol. 38, 100867, Apr. 2023.

[41] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Networks*, vol. 18, no. 5–6, pp. 602–610, Jul. 2005.

[42] A. Gharib, I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "An evaluation framework for intrusion detection dataset," in *Proc. 2016 International Conference on Information Science and Security (ICISS)*, 2016, pp. 1–6.

[43] A. Thakkar and R. Lohiya, "A review of the advancement in intrusion detection datasets," *Procedia Comput Sci*., vol. 167, pp. 636–645, 2020.

[44] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Proc. 2015 Military Communications and Information Systems Conference (MilCIS)*, 2015, pp. 1–6.

[45] N. Moustafa and J. Slay, "The evaluation of network anomaly detection systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set," *Information Security Journal: A Global Perspective*, vol. 25, no. 1–3, pp. 18–31, Apr. 2016.

[46] M. S. Al-Daweri, K. A. Zainol Ariffin, S. Abdullah, and M. F. E. M. Senan, "An analysis of the KDD99 and UNSW-NB15 datasets for the intrusion detection system," *Symmetry (Basel)*, vol. 12, no. 10, 1666, Oct. 2020.

[47] S. Naseer *et al*., "Enhanced network anomaly detection based on deep neural networks," *IEEE Access*, vol. 6, pp. 48231–48246, 2018.

[48] L. Mohammadpour, T. C. Ling, C. S. Liew, and A. Aryanfar, "A survey of CNN-based network intrusion detection," *Applied Sciences*, vol. 12, no. 16, 8162, Aug. 2022.