applied sciences

MDPI

Article

A Deep Learning Approach for Intrusion Detection Systems in Cloud Computing Environments

Wa'ad H. Aljuaid and Sultan S. Alshamrani *

Department Information Technology, College of Computers and Information Technology, Taif University, Taif 21944, Saudi Arabia; waad.otb19@gmail.com
* Correspondence: susamash@tu.edu.sa

Abstract: Cloud computing services have become indispensable to people's lives. Many of their activities are performed through cloud services, from small companies to large enterprises and individuals to government agencies. It has enabled clients to use companies' services on demand at the lowest cost anywhere, anytime, over the Internet. Despite these advantages, cloud networks are vulnerable to many types of attacks. However, as the adoption of cloud services accelerates, the risks associated with these services have also increased. For this reason, solutions have been implemented to improve cloud security, such as monitoring networks, the backbone of the cloud infrastructure, and detecting and classifying cyberattacks. Therefore, an intrusion detection system (IDS) is one of the essential defenses for detecting attacks in the cloud computing network. Current IDSs encounter some challenges in handling and simultaneously analyzing the large scale of traffic found in the cloud environment, and this affects the accuracy of cyberattack detection. Therefore, this research proposes a deep learning-based model by leveraging advanced convolutional neural networks (CNNs)-based model architecture to detect cyberattacks in the cloud environment efficiently. The proposed CNN-based model for intrusion detection consists of multiple significant stages: dataset collection, preprocessing, the SMOTE balance data strategy, feature selection, model training, testing, and performance evaluation. Experiments have demonstrated that the proposed model is highly effective in protecting cloud networks against various potential attacks. With over 98.67% accuracy, precision, and recall, the model has proven its ability to detect and classify network intrusions. Detailed analyses show that the model is proficient in securing cloud security measures and mitigating the risks associated with evolving security threats.

Keywords: cloud computing; intrusion detection system; CSE-CICIDS2018; deep learning

check for updates

Citation: Aljuaid, W.H.; Alshamrani, S.S. A Deep Learning Approach for Intrusion Detection Systems in Cloud Computing Environments. *Appl. Sci.* **2024**, *14*, 5381. https://doi.org/10.3390/app14135381

Academic Editor: Andrea Prati

Received: 25 April 2024
Revised: 12 June 2024
Accepted: 14 June 2024
Published: 21 June 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

In the world of information technology, cloud computing is now regarded as one of the greatest computing paradigms in recent years. The development of existing computer paradigms, such as parallel computing, grid computing, distributed computing, and cloud computing, can be considered the best among these models [1]. Cloud computing provides customers with a variety of computing resources and online storage. For both businesses and individuals, cloud services have recently become crucial [2]. In agreement with the National Institute of Standards and Technology (NIST), cloud computing is defined as "a model for convenient, on-demand network access to a shared pool of configurable computing resources (networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction" [3]. In order to better understand the security issues related to cloud computing and its services, we first need to understand the configuration of cloud computing. Based on NIST's description, "five essential characteristics, three service models, and four deployment models make up this cloud model", which will be discussed in the background section.

The security of cloud computing is of prime importance since it is used in many critical areas, such as education systems, healthcare systems, and e-government, which store important information. However, cloud computing potentially faces security threats if a network is compromised by an attacker [4]. Additionally, the Internet is utilized for providing access to cloud services, making it a significant source of vulnerabilities that can infect cloud resources and services. Consequently, improving cloud security becomes the primary challenge for cloud providers [5].

However, several cybersecurity methods, such as firewalls, antivirus software, data encryption algorithms, intrusion detection systems (IDSs), intrusion prevention systems (IPS), and others, have been used to provide security to the cloud environment from internal and external malicious attacks [6]. Although a variety of traditional mechanisms of defense are used to protect and detect attacks in the cloud-based network, there remains significant and continuous growth in threats and attacks due to the relentless evolution of cyberattacks [7]. Consequently, in order to increase security, this has resulted in a paradigm shift toward more intelligent and adaptive mechanisms, exemplified by machine learning (ML) and its subdomain, deep learning (DL), rather than traditional mechanisms of defense [8].

Intrusion detection systems (IDSs) in cloud computing earned a lot of attention during the past few years as one of the most effective ways to detect and recognize malicious activity [9]. In cybersecurity, an intruder is an entity that seeks to exploit system weaknesses [10]. Therefore, an IDS is used to detect an intrusion in a network, monitor all packets, and decide whether any of the incoming and outgoing packets have been affected by the intrusion [11]. Intrusion detection can be accomplished using one of two methods. The first involves traditional signature-based intrusion detection systems. However, these systems have limitations in detecting novel attacks and have not been suitable enough for large network scales such as cloud computing environments. Intelligent intrusion detection systems, on the other hand, use an anomaly-based approach based on learning methods [12]. This allows them to detect novel and unknown attacks by leveraging machine learning and deep learning models' ability to learn intricate traffic patterns and accurately pinpoint the nature of the attack [13].

Nowadays, the use of machine learning in cybersecurity has become popular, and cybersecurity researchers are relentless in ensuring that state-of-the-art cybersecurity mechanisms are achieved. However, with the increasing scale and complexity of the network, many machine learning-based detection techniques are not effective enough for learning large-scale network traffic data. Therefore, in recent years, deep learning-based detection techniques have gained more attention due to their ability to perform better with large-scale and complex network traffic and their ability to learn representations of features from raw data, making them adaptable to different attack scenarios [14].

The proposal contributions can be summarized as follows:

- Building an effective deep learning-based intrusion detection model in the cloud computing environment that covers a range of cyberattack scenarios.
- Utilizing the CNN paradigm for effective intrusion detection on the CSE-CICIDS2018 dataset [12] for evaluation.
- Implementing a multi-stage process to optimize detection accuracy and efficiency.
- Integrating advanced preprocessing techniques and feature selection methods, such as the Pearson correlation coefficient matrix heatmap, ensuring that the most relevant features are used for training.
- Selecting relevant features to reduce computational overhead and time complexity while enhancing the precision of the model.
- Training and testing the model in a cloud environment, demonstrating practical and scalable deployment.
- Addressing common issues such as data imbalance and feature redundancy, resulting in improved robustness and reliability.
- Providing significant advantages in terms of accuracy, efficiency, and scalability, positioning it as a superior solution for modern intrusion detection in cybersecurity.

2. Related Works

This section discusses different academic papers that proposed an intrusion detection system with various machine learning and deep learning approaches. The field of intrusion detection systems is widely studied in the literature. Some research papers, such as [15–17], cover the use of ML and DL to improve the network security of cloud computing.

Rawaa et al. [15] proposed an approach to improving the network intrusion detection system (NIDS) by optimizing deep learning (DL) models and employing binary particle swarm optimization (BPSO) as a feature selection method that improves the level of accuracy and time required to check traffic behavior. They evaluated this approach using a new and real dataset, the CSE-CICIDS2018 dataset, that covered all 11 criteria that determine the characteristics of a reliable dataset. Deep neural networks (DNNs), the most common deep learning algorithms, were used for training in this approach. The proposed approach contains three phases: The first phase is data preprocessing, which includes data cleaning, data digitalization, and data transformation. The second phase is feature selection, which uses the binary POS algorithm to determine the most relevant features. The experimental results of this approach achieved an accuracy of 95%.

In [16], the authors demonstrate a proposed real-time monitoring and detection "RT-AMD" of DDoS attacks on cloud computing using machine learning algorithms and an intrusion detection system, which uses anomaly-based detection. They selected machine learning classifiers of naïve Bayes (NB), decision tree (DT), k-neighbors (KN), and random forest (RF) to build a predictive model and compare them by two primary criteria: accuracy and detection rate.

In [17], the authors proposed a novel architecture that combines an autoencoder (AE) for feature learning with a deep neural network (DNN) for classifying network traffic into benign traffic or DDoS attack traffic. First, they performed data preprocessing to improve the quality of the input data and, subsequently, the quality and efficiency of the output, which includes label encoding, removing irrelevant features, and normalization. Then, the authors used a naïve (baseline) model using AE for feature extraction and DNN to classify DDoS attacks based on random hyperparameter values. After that, an optimized AE and DNN model was produced by adding improvements to this baseline model. The enhancements to the basic AE, such as unit norm, sparsity, orthogonality, and hyperparameter optimization using grid search, resulted in an optimized AE that produced effective representation and improved the classification. These significant features were fed into an improved DNN for classification, which was enhanced using optimization methods such as hyperband tuning of hyperparameters and intelligent learning rate determination. Experiments were performed on the NSL-KDD and CICIDS2017 datasets for validation, giving an accuracy of 98.43% and 98.92%, respectively.

The authors in [18] presented a hybrid IDS (HIDS) that combines signature-based intrusion detection (SIDS) and anomaly-based intrusion detection (AIDS) models based on machine learning algorithms. The NSL-KDD was used to evaluate this model. They developed a hybrid method using an ensemble of the C5.0 decision tree (signature) and one-class support vector machine (anomaly), which can result in a better detection rate than the single algorithms.

As with the same hybrid deep learning [19], the authors in another study proposed a hybrid intrusion detection system (IDS) based on hybrid deep learning that detects network attacks. The model is a convolutional recurrent neural network (CRNN) that combines an RNN and a CNN. The first stage used a CNN to collect local features; the second used a deep-layered RNN to capture features. They conducted an experiment, evaluated it using CICID2018 datasets, and produced an accuracy of 98.90%. An intrusion detection system (IDS) was developed in [9] based on the deep learning algorithm known as the deep forward neural network (DFNN). The dataset used in developing the model is the CICDDOS2019 dataset, which contains the most recent DDoS attacks. The proposed model was optimized using the Bat optimization algorithm and had high evaluation

metrics compared to RNN, LSTM, and ensemble models. The model achieved an accuracy of 98.87%.

Ahmet and Zafer [20] suggested an approach of an intrusion detection system based on mixing two deep learning algorithms, convolutional neural network (CNN) and long short-term memory (LSTM), known as the hybrid method. This approach first used CNN layers for feature extraction from the input automatically and then used the LSTM for predicting sequence, and it has seven layers to achieve high performance and accuracy when compared with each CNN and LSTM individually. Therefore, evaluation criteria were applied to the NSL-KDD dataset. The evaluation of the results of four criteria, which were accuracy, precision, recall, and F1-score, showed that the achieved accuracy was 99.20%.

The authors [21] proposed a deep learning approach for intrusion detection using a recurrent neural network (RNN-IDS). They used the NSL-KDD dataset to evaluate their performance in detecting network intrusion. The performance of this model is in binary classification, which identifies network traffic behavior as normal or anomalous, and multiclass classification, which classifies network traffic into five categories, which are normal or any one of the other four attack types: user-to-root (U2R), probe (Probing), denial of service (DOS), and root to local (R2L). The results compared with other traditional machine learning methods in terms of accuracy showed that RNN-IDS outperformed them.

The authors [22] constructed a novel network intrusion system (NIDS) based on convolutional neural networks (CNNs). They trained deep learning based on detection models using extracted features and original network traffic. This model achieved a detection accuracy of 96.55% using the feature dataset of CICID 2017. The results showed that this CNN-based model outperforms SVM and DBN in accuracy.

The authors in [23] presented an efficient method to detect DDoS attacks using anomaly detection by the DL algorithm, which was long short-term memory (LSTM). The LSTM is a specialized form of a recurrent neural network (RNN) that solves the problem of vanishing and exploding gradients with decreased training time and increased accuracy. This algorithm was trained and tested on the CICIDS2017 dataset, consisting of numerous attack types. The paper focused on four layers: data collection, feature identification, model training, and execution of the classification model. The proposed training model contains one input layer of LSTM with 77 neurons and a tanh activation function. Then, it has 12 hidden layers, 5 of which are LSTM, and 7 others are fully connected layers. Finally, the output layer has two fully connected neurons with a sigmoid activation function. These 14 recurrent neural network model layers provided an accuracy of 96.25%.

Fahimeh and Jukka [24] presented a deep learning approach for an intrusion detection system (IDS) based on a deep autoencoder (DAE), which is one of the most well-known deep learning models. The architecture of this model consists of four autoencoders, where the output of each autoencoder is used as input to the next autoencoder. In addition, they used unsupervised feature learning in the training phase, which is a greedy layer-wise pre-training that involves training each layer of the neural network as a separate autoencoder and that helps to improve the deep model performance and supervised fine-tuning. After training four autoencoders, they used the SoftMax classifier to classify the attack classes from the input dataset. They performed numerous experiments on the KDD-CUP'99 dataset, and the proposed approach achieved a detection accuracy of 94.71%.

Another study [25] mentioned developing a model for detecting DDoS attacks in cloud environments using a supervised learning algorithm based on the intrusion detection system (IDS). They proposed two approaches: the filter method represented by learning vector quantization (LVQ) and the dimensionality reduction method defined by principal component analysis (PCA), and the dataset was collected from NSL-KDD. The selected features from each approach were used for classification, and the classifiers were naïve Bayes (NB), support vector machine (SVM), and decision tree (DT). Based on the experiment, LVQ, for feature selection with the DT model, identified the attacks more accurately than other methods.

The authors evaluated this model using the DDoS-2020 and NSL-KDD datasets. The results of this proposed model achieved high accuracy with the random forest algorithm, reaching 99.38% with the DDoS-2020 dataset and 99.30% with the NSL-KDD dataset.

An intrusion detection system (IDS) was proposed [26], implemented, and trained using different deep neural network architectures, including convolutional neural networks, autoencoders, recurrent neural networks, and LSTM. These deep learning models were trained on NSL-KDD training datasets and evaluated on both datasets provided by NSL-KDD for testing. They used conventional ML IDS models with different well-known classification techniques, including decision tree, KNN, random forest, support vector machine, and naïve Bayes, to compare with deep neural network models.

The authors in [27] developed an intrusion detection system (IDS) utilizing a deep learning (DL) algorithm, specifically a multi-layer perceptron (MLP) neural network, trained and tested on the KDDCup99 dataset. The detection process involved several steps to maximize accuracy and minimize loss. Initially, the KDDCup99 dataset was collected and preprocessed using techniques like one-hot encoding and feature scaling (standardization). The preprocessed data were then used to create a pre-trained MLP model with two layers: an input layer using the RELU activation function and an output layer using the SoftMax activation function. The loss function was cross-entropy, optimized with the Adam optimizer. The output layer has five nodes representing the following attack classes: DoS, probe, R2L, U2R, and normal. The experiments demonstrated that the model achieved an accuracy of 91.4%.

The authors of [28] proposed an adaptive ensemble learning model for the field of intrusion detection systems (IDSs). The primary purpose of using an ensemble learning model is to integrate the advantages of each algorithm for different types of data detection and achieve optimal results through ensemble learning. This model used the NSL-KDD dataset and some common machine learning algorithms such as decision trees, random forests, SVMs (support vector machines), logical regression, k-nearest neighbors (KNN), Adaboost, and deep neural networks to train it. Then, an adaptive voting algorithm with different class weights was used to obtain the optimal detection results. They used the NSL-KDD Test+ dataset to verify their approach, and the accuracy of the proposed adaptive voting algorithm was 85.2%.

Weikai Ren, Ningde Jin, and Lei OuYang [29] introduced the PSGCN (phase space graph convolutional network) model, which uses deep learning to analyze complex networks by converting time series data into graph structures. This approach is relevant to IDS-based CNN algorithms in cloud environments, where CNNs analyze network traffic patterns to detect intrusions. The comparison highlights the potential of advanced deep learning models to enhance the accuracy and robustness of complex network analyses.

Table 1 summarizes various ML and DL approaches for intrusion detection systems with different datasets. The next table shows that some researchers provide highly efficient results using DL for intrusion detection systems.

Table 1. Comparison of the latest research focusing on ML for IDS.

Reference	Year	Dataset	Learning Algorithm	Accuracy
[15]	2020	CSE-CICIDS2018	Deep neural networks (DNNs)	95%
[16]	2022	DDoS-2020 NSL-KDD	Naïve Bayes, decision tree, k-neighbors, and random forest.	99.30%
[17]	2020	NSL-KDD CICIDS2017	Combines optimization AE and DNN	98.43% 98.92%
[18]	2020	NSL-KDD	Ensemble technique of C5.0 Decision tree and support vector machine	83.24%
[19]	2023	CICID2018	Hybrid deep learning techniques based on RNN and CNN	98.90%

Table 1. *Cont.*

Reference	Year	Dataset	Learning Algorithm	Accuracy
[20]	2023	NSL-KDD	Hybrid method based on CNN and LSTM	99.20%
[21]	2017	NSL-KDD	Recurrent neural network (RNN)	97%
[22]	2020	CICID2017	Convolutional neural network (CNN)	96.55%
[23]	2020	CICID2017	Long short-term memory (LSTM)	97%
[24]	2018	KDD-CUP'99	Deep autoencoder (DAE)	94.71%
[25]	2020	NSL-KDD	Naïve bayes, support vector machine, and decision Tree.	98%
[26]	2018	NSL-KDD	Deep neural network models including DCNN, LSTM, and autoencoders	85% 89%
[27]	2020	Kddcup99	Multi-layer perceptron (MLP)	91.4%
[28]	2019	NSL-KDD	Ensemble learning model with various ML algorithms	85.2%

3. Materials and Methods

3.1. Proposed Model

The proposed model introduces an efficient application of the convolutional neural network (CNN) paradigm for intrusion detection, utilizing the CSE-CICIDS2018 dataset. The proposed approach for intrusion detection consists of seven major stages, as shown in the following Figure 1.

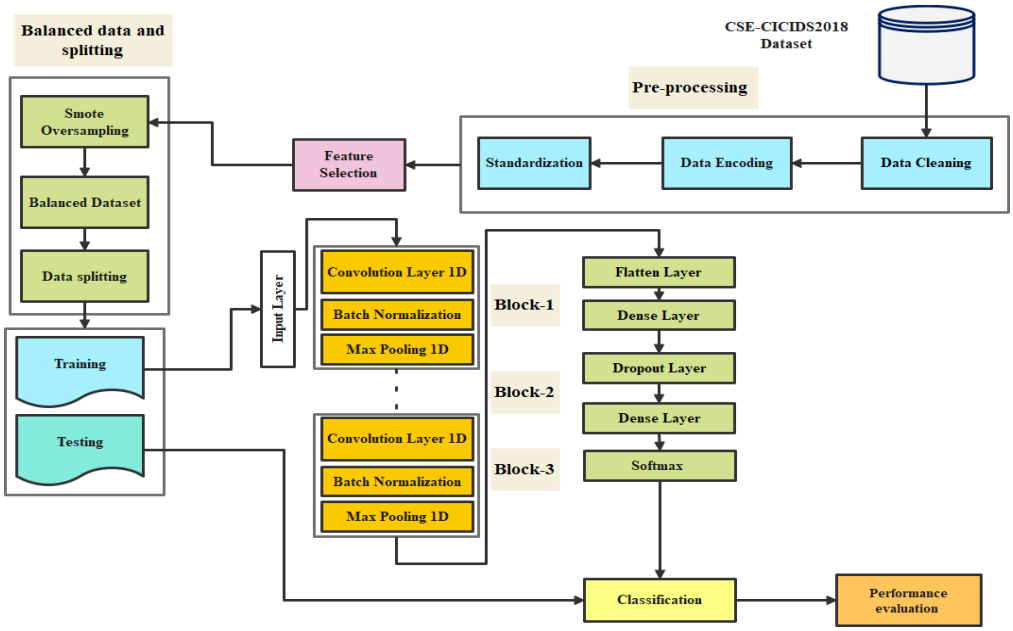


Figure 1. The proposed model.

The dataset collection is the first stage in which we acquired the dataset based on which we trained and tested the performance of deep learning models for intrusion detection in the cloud environment. In the second stage, we preprocessed the collected dataset to remove missing or null values and performed label encoding and data standardization. In the third stage, we applied a feature selection technique based on a Pearson correlation coefficient matrix heatmap to efficiently select useful features to train the deep learning models.

In the fourth stage, we balanced the dataset with SMOTE oversampling and undersampling and then split the dataset into training and testing sets. In the fifth and sixth stages, we trained and tested deep learning models for the detection and classification of cyberat-

tacks. Finally, we evaluated the performance of trained deep learning models and proposed the best-performing model for efficient intrusion detection in the cloud environment.

The proposed intrusion detection model for cloud environments leverages a convolutional neural network (CNN) architecture with multiple layers and blocks to efficiently detect intrusions. The model features three Conv1D layers, each with 64 filters and a kernel size of 6 and activated by the ReLU function, followed by batch normalization for stable training and MaxPooling1D layers for down-sampling. After the convolutional and pooling layers, a Flatten layer converts the feature maps into a one-dimensional vector, which is then processed by two fully connected Dense layers with 64 units each and ReLU activation. A Dropout layer with a 0.5 dropout rate is included to prevent overfitting. The final output layer is a Dense layer activated by the SoftMax function to generate class probabilities. Our CNN parameters are shown in Table 2.

Table 2. Parameter of proposed model.

Parameter	Value
Epochs	10
Batch size	64
Activation function	ReLu, SoftMax
Loss function	Categorical cross-entropy
Optimizer	Adam

During training, the preprocessed data are divided into balanced sets for training and testing, with the model iteratively extracting hierarchical representations of the input data through the Conv1D layers, batch normalization, and MaxPooling layers. The Dense layers, positioned after the Flatten layer, consolidate the extracted features for classification, with the first Dense layer acting as an intermediary and the Dropout layer enhancing generalization. The final Dense layer before the Softmax activation outputs probability distributions for each class, providing predictive uncertainty. Post-training evaluation on an independent dataset assesses the model's real-world performance using metrics such as accuracy, precision, recall, and F1-score, demonstrating the model's reliability and effectiveness in network intrusion detection for cloud environments. All of these stages are further described in the following subsections.

3.2. Data Collection

One of the most important considerations that comes with an intrusion detection system is choosing an appropriate dataset, and it is a critical task. As network attacks evolve with time, the use of old datasets may not lead to objective and desired results. Thus, we chose to use one of the recently published IDS datasets that were generated on the AWS (Amazon Web Services) cloud environment, which is the CSE-CICIDS2018 dataset [30]. It is appropriate for our research objectives as it includes both normal cloud network traffic as well as up-to-date attacks traffic, and it is publicly available in [30].

The CES-CICIDS2018 dataset consists of seven distinct scenarios simulating various network intrusions, crucial for evaluating the effectiveness of intrusion detection models. Each scenario includes a mix of benign and cyberattack instances, providing a diverse testing environment as shown in Table 3.

Each scenario is designed to test specific types of attacks and benign activities, ensuring the comprehensive evaluation and improvement of detection algorithms. This structured approach enhances the model's accuracy and robustness, enabling it to address both common and sophisticated network threats effectively.

Table 3. All seven scenarios.

Scenarios	Contents
Scenario 1: Brute Force Attack Simulation	Benign: 667,626 instances FTP Brute Force: 193,360 instances SSH Brute Force: 187,589 instances
Scenario 2: DoS Attack Types	Benign: 996,077 instances DoS GoldenEye: 41,508 instances DoS Slowloris: 10,990 instances
Scenario 3: DDoS LOIC-HTTP	Benign: 7,372,557 instances DDoS attacks-LOIC-HTTP: 576,191 instances
Scenario 4: Advanced DDoS Techniques	Benign: 686,012 instances DDOS attack-LOIC-UDP: 360,833 instances DDOS attack-HOIC: 1730 instances
Scenario 5: Web Application Attacks	Benign: 1,048,213 instances Brute Force-Web: 249 instances Brute Force-XSS: 79 SQL Injection: 34 instances
Scenario 6: Infiltration Tactics	Benign: 1,048,009 instances Infiltration: 151 instances
Scenario 7: Botnet Detection	Benign: 762,384 instances Bot: 286,191 instances

3.3. Data Preprocessing

Data preprocessing is an essential phase in deep learning processes to improve the quality of the data and, subsequently, the quality of the output and efficiency of training and testing the model. When dealing with a large dataset, the data preprocessing techniques can be utilized to deal with multiple issues at once [20]. The dataset for this study contains redundant records, null and outlier values, and irrelevant and categorical features. Therefore, we needed to filter and clear the CES-CICIDS2018 dataset [30] and select the most relevant samples. Table 4 presents categories of attacks and the number of samples for each attack. In dataset preprocessing, we performed data cleaning, data standardization, and label encoding, which are described in the following subsections.

Table 4. Attack classes of CES-CICIDS2018 dataset.

Class	Traffic Type	Count
Normal	Normal	13,484,708
DDoS	Attack (HOIC, LOIC-UDP, and LOIC-HTTP)	1,263,933
DoS	Attack (Hulk, GoldenEye, Slowloris, and SlowHTTPTest)	654,300
Brute Force	Attack (FTP and SSH)	380,949
Bot	Attack (Botnet)	286,191
Infiltration	Attack (Infiltration)	161,934
Web	Attack (Web, XSS, and SQL Injection)	928

3.3.1. Data Cleaning

In the data cleaning step, we removed the missing, null, and infinity values from the dataset as they transfer wrong data to the deep learning model, which can result in a decrease in the accuracy of the model. Therefore, we removed such irrelevant samples that are non-informative or invalid such as null and infinity values.

3.3.2. Label Encoding

Label encoding is a technique that is used to transform categorical feature values to numerical values so that they can be fitted by deep learning models that only accept numerical data. The features can be encoded by many methods. Among them, one-hot

encoding is the most popular method; thereby, we applied one-hot encoding to convert the categorial feature values into numerical values.

3.3.3. Data Standardization

Data standardization is one of the most important parts of preprocessing techniques. Standardization is a rescaling of feature values to ensure the standard deviation becomes 1 and the mean value becomes 0. The standardization reduces all of the data's features to a common scale. The main purpose of using standardized data to train deep learning models is to learn faster when all features are on the same scale. In standardization, the average value of each feature in the dataset is subtracted from each value and then divided by the standard deviation. In this way, data are normally distributed. Usually, standardization is performed using the Z-score formula given below in Equation (1):

$$Z = (x - \mu)/\sigma \quad (1)$$

Equation (1) is the standard equation of standardization, where μ is the mean of the data and σ is the standard deviation of the data.

3.4. Fix Imbalance Class

The sample count of different classes in the CES-CICIDS2018 dataset [15] is listed in Table 4. The same sample distribution of different classes in the CES-CICIDS2018 dataset [15] is visualized in Figure 2. It can be clearly seen that there is a class imbalance issue in this dataset, i.e., the normal class has a much higher proportion of samples as compared to other class samples. Therefore, to fix this issue, we needed to apply an under-sampling technique at first to reduce the normal class samples.

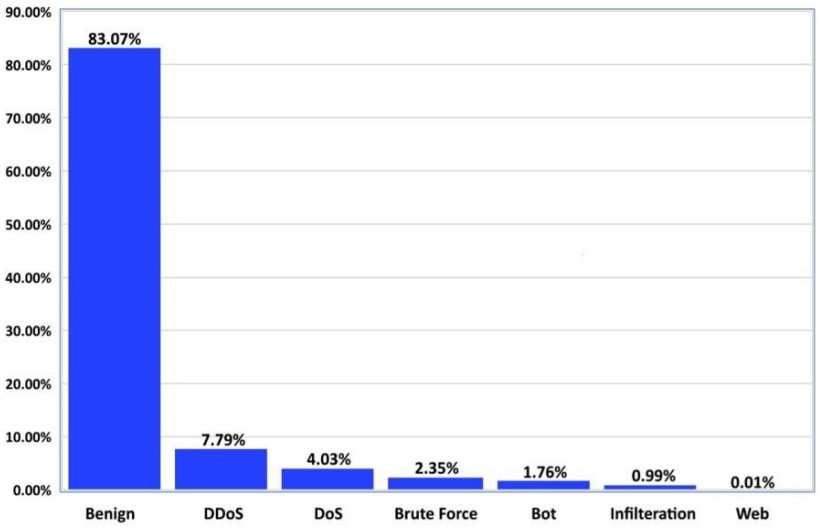


Figure 2. Distribution for each class in the CES-CICIDS2018 dataset.

Applying the under-sampling technique would not have completely fixed the class imbalance issue since the other classes also have an imbalance issue. Therefore, after under-sampling the normal class traffic, we applied the synthetic minority oversampling technique (SMOTE) to fix the class imbalance issue [31].

3.5. Dataset Splitting

In order to train and test deep learning models, we need to split the data into a train set and a test set with a ratio of 70:30, i.e., 70% of the dataset is randomly selected for training and 30% for testing. During the training phase, only the training set is exposed to the deep learning model for its training, and the test set is only passed through deep learning after the training is complete. In this way, we can analyze the performance of the

trained model over unseen data to confirm if there is an under-fitting or overfitting issue in a model's training.

3.6. Feature Selection

As discussed earlier, the CES-CICIDS2018 dataset [29] has over 80 features. The large number of input features significantly influences the performance of the deep learning model. Feature selection plays a crucial part in a deep learning model's success. Feature selection is also a pivotal element in the effectiveness of any intrusion detection system (IDS) approach. Its impact is manifold, including reducing model complexity, which minimizes computational costs. Additionally, it simplifies model debugging, thus enhancing the model's interpretation of the learning results. Further, it massively reduces the model training time and improves the training accuracy [32]. We applied Pearson correlation matrix analysis to the CES-CICIDS2018 dataset to calculate the correlation matrix heatmap. This helps identify features that are highly correlated with each other, allowing one to make informed decisions about which features to potentially drop from the data.

3.7. Model Training

After all of the preprocessing and feature selection steps, we split the dataset into a training and test set. Then, we passed the training data to the CNN blocks model to detect intrusion in the cloud environment. The features obtained from the preprocessing stage were given to the deep learning model for training and testing. In the training phase, the data or features obtained from the preprocessing stage were given to the model, which started training on these features. After the training phase, the model was tested to assess its performance, which helped obtain better results. Sometimes, hyperparameter tuning was required if the desired results were not obtained.

3.8. Model Testing and Performance Comparison

After the training phase, we needed to evaluate the performance of models based on some parameters. For classification problems, accuracy, precision, recall, and F1-score were used to assess the model's performance. We passed the test set to all trained models and evaluated their performance individually based on the above-listed parameters. After calculating the performance score for CNN-based model architecture, we needed to compare the performance of the proposed model with recent works.

4. Experimental Result

In this section, we delve into the findings that highlight the remarkable accuracy of a deep learning model based on the CNN blocks model in classifying network traffic. The study employs rigorous data analysis techniques, specifically convolutional neural network modeling, to provide a solid foundation for data interpretation. The reshaped CSE-CICIDS2018 dataset, consisting of seven data files divided into different scenarios, was meticulously formatted for CNN input. The model underwent thorough training and analysis to ensure optimal performance.

The performance of the model was evaluated by differentiating between various categories of network traffic. Performance metrics were then calculated for each scenario. The resulting confusion matrix, precision, recall, F1-score, and overall accuracy were significantly high, which indicates an effective classification.

4.1. CSE-CICIDS2018 Scenario 1 Results

Upon studying the CSE-CICIDS2018 dataset, the initial scenario presents a captivating classification challenge containing "Benign", "FTP-BruteForce", and "SSH-Bruteforce". This research analyzes the efficacy of a model trained in the characteristics of these unique traffic types, as illustrated in Tables 5 and 6.

The results in Table 5 present an overview of the model's performance, providing a comprehensive understanding of its effectiveness. With an accuracy rate of 99.937%,

matched by precision, recall, and F1-score, all at the same value, the model's consistency and reliability in classifying diverse network traffic types are affirmed.

Table 5. The evaluation performance of scenario 1 in the proposed model per category.

Category	Precision	Recall	F1-Score
Benign	0.99995	0.99836	0.99915
Ftp Bruteforce	0.99919	1	0.99959
Ssh Bruteforce	0.99896	0.99975	0.99936

Table 6. The overall evaluation performance of scenario 1 in the proposed model.

Metrics	Results
Accuracy	0.99937
Precision	0.99937
Recall	0.99937
F1-score	0.99937

4.2. CSE-CICIDS2018 Scenario 2 Results

Further investigation of the CSE-CICIDS2018 dataset reveals scenario 2, which poses a complex classification challenge involving "Benign", "DoS_GoldenEye", and "DoS_Slowloris" traffic. This section explores the model's ability to effectively differentiate and categorize these distinct traffic types, bringing on the analytical findings presented in Tables 7 and 8.

Table 7. The evaluation performance of scenario 2 in the proposed model per category.

Category	Precision	Recall	F1-Score
Benign	0.99965	0.999	0.99933
DoS_GoldenEye	0.99925	0.9997	0.99947
DoS_Slowloris	0.9995	0.9997	0.9996

Table 8. Evaluation classification of scenario 2 results of the proposed model.

Metrics	Results
Accuracy	0.99947
Precision	0.99947
Recall	0.99947
F1-Score	0.99947

In scenario 2, Table 8 provides an overview of the model's performance metrics, showcasing its accuracy of 99.947% and consistent precision, recall, and F1-score values. These metrics demonstrate the model's reliable and stable classification capabilities in various network traffic types, indicating a nuanced understanding and effective detection ability in this scenario.

4.3. CSE-CICIDS2018 Scenario 3 Results

In the CSE-CICIDS2018 dataset, scenario 3 requires differentiation between "Benign" traffic and "Ddos Attacks-LOIC-HTTP". This analysis evaluates the model's ability to categorize these traffic types. We draw insights from Tables 9 and 10.

Table 9. The evaluation performance of scenario 3 in the proposed model per category.

Category	Precision	Recall	F1-Score
Benign	1	0.98527	0.99258
Ddos Attacks-LOIC-HTTP	0.98573	1	0.99282

Table 10. Overall classification results of scenario 3 of the proposed model.

Metrics	Results
Accuracy	0.9927
Precision	0.9928
Recall	0.9927
F1-Score	0.9927

Table 9 assesses the model's performance in scenario 3. It provides a detailed account of each category's precision, recall, and F1-score. The model demonstrates perfect precision (1.00) in identifying "Benign" traffic with a recall of 98.527%, resulting in an F1-score of 99.258%. These metrics are proof of the model's remarkable accuracy and reliability in recognizing benign activities without misclassification.

Table 10 summarizes the overall performance metrics for scenario 3. From Table 10, it can be seen that scenario 3 resulted in an accuracy rate of 99.27% and consistent precision, recall, and F1-score values of around 99.28%.

4.4. CSE-CICIDS2018 Scenario 4 Results

Scenario 4 of the CSE-CICIDS2018 dataset delves into the model's ability to differentiate between "Benign" traffic and two types of DDoS attacks: "DDoS Attacks-LOIC-UDP" and "DDoS Attacks-HOIC". This analysis aims to assess the model's accuracy and efficiency in categorizing these various types of traffic. To gain insights, we reference Tables 11 and 12, which provide a visual representation. Table 11 comprehensively evaluates the model's performance in scenario 4. It highlights the model's exceptional precision, recall, and F1-score across all categories.

Table 11. The evaluation performance of scenario 4 in the proposed model per category.

Category	Precision	Recall	F1-Score
Benign	1	1	1
DDOS Attack-LOIC-UDP	1	1	1
DDOS Attack-HOIC	1	1	1

Table 12. Overall classification results of scenario 4 in the proposed model.

Metrics	Results
Accuracy	1
Precision	1
Recall	1
F1-Score	1

The impeccable scores achieved in all metrics demonstrate the model's high accuracy in detecting benign and malicious traffic, highlighting its efficacy in a controlled environment. This level of precision is vital in ensuring that network security measures can proactively thwart such attacks without mistakenly flagging benign activities. Table 12 summarizes the comprehensive performance metrics for scenario 4, highlighting the model's impeccable classification capabilities.

4.5. CSE-CICIDS2018 Scenario 5 Results

Within the CSE-CICIDS2018 dataset, there is a designated scenario 5 that aims to test a model's capability to differentiate between "Benign" traffic and a range of cyberattacks: "Brute Force-Web", "Brute Force-XSS", and "SQL Injection". The current analysis measures the model's accuracy and effectiveness in detecting and classifying these threats. We obtain insights from Tables 13 and 14. Table 13 evaluates the model's ability to identify and categorize each type of network traffic.

Table 13. The evaluation performance of scenario 5 in the proposed model per category.

Category	Precision	Recall	F1-Score
Benign	0.9921	0.9825	0.9873
Brute Force-Web	0.9923	0.9928	0.9923
Brute Force-XSS	0.9931	0.9931	0.9931
SQL Injection	1	1	1

Table 14. Overall classification results of scenario 5 in the proposed model.

Metrics	Results
Accuracy	0.9929
Precision	0.994
Recall	0.99175
F1-Score	0.993

The model's results demonstrate exceptional precision when detecting "Benign" traffic, achieving an almost flawless recall and F1-score. It indicates that the model is highly reliable in avoiding false positives. And, therefore, the model performs well with "Brute Force-Web", "Brute Force-XSS", and "SQL Injection", exhibiting high recall rates, critical for preventing potential security breaches caused by these more sophisticated attacks. Table 14 summarizes the overall performance metrics for scenario 5, providing insight into the model's general classification abilities across various traffic types.

4.6. CSE-CICIDS2018 Scenario 6 Results

Scenario 6 in the CSE-CICIDS2018 dataset presents a unique challenge of distinguishing between "Benign" traffic and a range of malicious activities categorized under "Label" and "Infiltration". The model's accuracy is evaluated through precision, recall, and F1-scores for these categories, illustrated in Tables 15 and 16. Table 15 provides a comprehensive analysis of the model's performance in classifying each type of attack and normal traffic for scenario 6.

Table 15. The evaluation performance of scenario 6 in the proposed model per category.

Category	Precision	Recall	F1-Score
Benign	1	0.92941	0.96341
Label	0.84068	0.96145	0.89702
Infiltration	0.95194	0.88212	0.9157

Table 16. Overall classification results of scenario 6 in the proposed model.

Metrics	Results
Accuracy	0.92432
Precision	0.93078
F1-Score	0.92532

The model consistently displays remarkable precision in accurately identifying "Benign" traffic, with a high recall and F1-score reducing the occurrence of false positives. However, precision is comparatively lower for "Label", which results in more false positives in this category. Conversely, the model exhibits robust precision for "Infiltration" attacks, although the recall is lower, implying the possibility of some false negatives and potential oversight of actual infiltration attempts.

Table 16 presents a comprehensive overview of the performance metrics for scenario 6, highlighting a dependable and efficient classification approach for diverse traffic types.

4.7. CSE-CICIDS2018 Scenario 7 Results

In scenario 7 of the CSE-CICIDS2018 dataset, the model was challenged to differentiate between "Benign" traffic and "Bot" attacks, evaluating its accuracy and ability to classify each network traffic type effectively. The results of this evaluation are presented in Tables 17 and 18.

Table 17. The evaluation performance of scenario 7 in the proposed model per category.

Category	Precision	Recall	F1-Score	Support
Benign	1	1	1	20078
Bot	1	1	1	19922

Table 18. Overall classification results of scenario 7 in the proposed model.

Metrics	Results
Accuracy	1
Precision	1
Recall	1
F1-Score	1

Table 16 provides detailed performance metrics for each category in scenario 7, revealing exceptional precision, recall, and F1-score scores for both "Benign" and "Bot" traffic. These results indicate the model's remarkable capability to distinguish between normal and malicious traffic without misclassification.

Table 18 summarizes the model's overall performance metrics for scenario 7, strengthening its exemplary classification capabilities.

4.8. Proposed Model Architecture and Space Complexity

We calculated the complexity of this CNN model by analyzing both the parameter count and the number of floating-point operations (FLOPs) as shown in Table 19. The parameter count is determined by summing the weights and biases in each layer, including Conv1D, Dense, and batch normalization layers. The FLOPs are computed based on the number of multiplications and additions required by the convolutional and Dense layers to process the input data. By aggregating these metrics, we assessed the model's computational and memory requirements.

Table 19. Proposed model architecture and space complexity.

Layer	Output Shape	Parameters
Conv1D	76 × 64	448
Batch normalization	76 × 64	256
MaxPooling1D	38 × 64	0
Conv1D	38 × 64	24,640
Batch normalization	38 × 64	256
MaxPooling1D	19 × 64	0
Conv1D	19 × 64	24,640
Batch normalization	19 × 64	256
MaxPooling1D	10 × 64	0
Flatten	1 × 640	0
Dense	1 × 64	41,024
Dropout	1 × 64	0
Dense	1 × 64	4160
Dense	1 × 2	130

Table 19. *Cont.*

Layer	Output Shape	Parameters
Total params	95,810	Memory Size: 374.26 KB
Trainable params	95,426	Memory Size: 372.76 KB
Non-trainable params	384	Memory Size: 1.50 KB
Total FLOPs	559,507	

The CNN model has a moderate complexity with 95,810 parameters and 559,507 FLOPs, striking a balance between computational efficiency and learning capacity. Its architecture includes three Conv1D layers, each followed by batch normalization and max pooling, reducing the spatial dimensions while preserving essential features. The model's parameter count is concentrated in the Conv1D and Dense layers, with a total memory footprint of 374.26 KB, making it suitable for deployment on devices with limited resources. This design ensures effective training and inference for sequential data tasks while maintaining manageable computational and memory requirements.

5. Discussion

Various experimental scenarios have demonstrated the suitability of using CNNs for IDS in cloud environments. With their exceptional feature extraction and ability to learn complex data without manual intervention, and because of their ability to process large-scale and high-dimensional data, CNNs are pivotal in identifying subtle signs of cyberattacks.

Table 20 provides performance metrics for all seven cyber scenarios to illustrate the model's effectiveness across different types of cyberattacks. The scenarios encompass a range of attacks, from brute force attack simulations and DoS attack types to advanced DDoS techniques, web application attacks, infiltration tactics, and botnet detection. Notably, the model demonstrates near-perfect accuracy, precision, recall, and F1-scores in most scenarios, such as scenario 1 ("Brute Force Attack Simulation") (99.94%) and scenario 2 ("DoS Attack Types") (99.95%). Particularly impressive are the perfect scores in scenario 4 ("Advanced DDoS Techniques") and scenario 7 ("Botnet Detection"), each achieving 100% across all metrics, indicating flawless identification and classification of these sophisticated threats. In contrast, scenario 3 ("DdoS LOIC-HTTP") shows slightly lower, but still highly commendable, metrics with an accuracy of 99.27%. This suggests a minimal, yet existent, margin for error. Similarly, scenario 5, ("Web Application Attacks") exhibits high performance, with an accuracy of 99.29% and a notable precision of 99.4%, slightly higher than its recall of 99.18%, pointing toward a strong detection rate with minimal false positives. The most significant drop in performance is observed in scenario 6 ("Infiltration Tactics"), where the accuracy is 92.43% and the precision is slightly higher at 93.08%. This indicates a relatively greater challenge in accurately identifying infiltration tactics compared to other attack types, though the performance remains robust.

After analyzing various scenarios, it has been found that CNNs are highly effective in detecting threats in cloud computing environments. The model has demonstrated consistently high precision and recall values, especially for complex and sophisticated attacks. Its ability to adapt to attacks and its reliability in real-time attack detection and classification are noteworthy. Table 18 presents the evaluation results for all seven scenarios of the proposed model, along with the average performance across the entire model for multi-classification using a CNN-based deep learning approach on the CSE-CICIDS2018 dataset within a cloud computing environment.

Table 20. The overall evaluation performance of all 7 scenarios of the proposed model.

Scenarios	Accuracy	Precision	Recall	F1-Score
Scenario 1: Brute Force Attack Simulation	99.94%	99.94%	99.94%	99.94%
Scenario 2: DoS Attack Types	99.95%	99.95%	99.95%	99.95%
Scenario 3: DDoS LOIC-HTTP	99.27%	99.28%	99.27%	99.27%
Scenario 4: Advanced DDoS Techniques	100%	100%	100%	100%
Scenario 5: Web Application Attacks	99.29%	99.4%	99.18%	99.3%
Scenario 6: Infiltration Tactics	92.43%	93.08%	92.43%	92.43%
Scenario 7: Botnet Detection	100%	100%	100%	100%
Average Results	98.69%	98.80%	99.68%	98.79%

Comparison with the State-of-the-Art

During this discussion, it was found that applying CNNs to IDSs in cloud computing environments using the CSE-CICIDS2018 dataset has led to a significant advancement in cybersecurity measures. It highlights the superior performance of the proposed multi-blocks of the CNN model, which includes some strategies such as the SMOTE balance data strategy to avoid imbalance class issues and using a Dropout layer to mitigate the overfitting problem. This model achieved an exceptional accuracy of 98.67%. Compared to other models listed in Table 1, our approach effectively manages common issues such as data imbalance and feature redundancy, leading to improved robustness and reliability. Moreover, utilizing CNNs allows our model to capture complex patterns and anomalies in the dataset, outperforming traditional and hybrid methods that lack this capability. Our integrated and methodical approach provides significant advantages in accuracy, efficiency, and scalability, making it a superior solution for intrusion detection in modern cybersecurity landscapes.

Table 21 lists other research that has dealt with IDSs and used the same CSE-CICIDS2018 dataset with different deep learning approaches. It is important to note that the approaches resulted in slightly lower accuracies for models such as fully connected Dense DNN [15,33] and autoencoder [34], which achieved 90%, 95%, and 95.77%, respectively. These single-model applications usually struggle with complex and imbalanced datasets. However, the study's approach of combining robust data augmentation techniques with a hybrid deep learning model provides a significant improvement over traditional single-model applications, such as those existing in MLP with BP [35] and PCA-DNN [36], which achieved higher accuracies of 98.41% and 97.77%, respectively. The impact and operation of our proposed model for intrusion detection are significant, reflecting advancements in both methodological rigor and practical applicability. By employing the CNN paradigm, our model effectively identifies and mitigates network intrusions with high accuracy. Compared to the single-block CNN architecture in [37], which achieved 98.15% accuracy, our three-block CNN model significantly enhances feature extraction and performance, achieving a higher accuracy of 98.67%. The operation begins with the meticulous collection and preprocessing of the CSE-CICIDS2018 dataset, ensuring the data are clean and standardized, which is necessary for the integrity of the training process, separated into seven scenarios. Following this, the Pearson correlation coefficient matrix heatmap is utilized to select the most relevant features, optimizing the model's performance and reducing computational complexity. Our CNN model's ability to process and analyze high-dimensional data allows for the detection of intricate intrusion patterns that traditional methods might miss. Its deployment in a cloud environment highlights scalability and real-world applicability, making it capable of handling large-scale network data and evolving threats. It leads to superior detection rates, reduces

false positives, and contributes to enhanced cybersecurity measures. Compared to existing models, our approach balances accuracy, efficiency, and operational practicality, making it a robust tool for modern intrusion detection systems.

Table 21. Comparison of the latest research on DL for IDSs based on the CES-CICIDS2018 dataset.

Reference	Year	Learning Algorithm	Accuracy
[15]	2020	DNN	95%
[33]	2020	Fully Connected Dense DNN	90%
[34]	2021	Autoencoder	95.79%
[35]	2023	MLP with BP	98.41%
[36]	2022	PCA-DNN	97.77%
[37]	2022	CNN	98.15%
Proposed	2024	Multi-Blocks of CNN	98.67%

6. Conclusions and Future Work

Cloud computing has revolutionized the field of information technology with its vast applications. However, despite the use of various cybersecurity methods, cyberattacks in cloud environments are on the rise. To ensure the security of these systems, it is essential to find solutions and measures to reduce these attacks. One of the critical cybersecurity defenses against cyberattacks in a cloud computing environment is an intrusion detection system (IDS). However, current IDSs must be equipped to handle the vast amounts of analyzed data and real-time detection, reducing their performance. Therefore, in our work, we propose a deep learning model to improve the security of cloud computing environments by detecting and classifying cyberattacks. This research thoroughly examined a deep learning model based on CNNs for network traffic classification in cloud computing environments. With cyber threats becoming increasingly sophisticated, security poses significant challenges. By utilizing the CSE-CICIDS2018 dataset in various scenarios, the study demonstrated the model's exceptional ability to effectively differentiate between benign and malicious traffic with high accuracy. These findings highlight the potential of advanced deep learning techniques to significantly enhance traditional IDSs, which are crucial for securing cloud environments.

The implications of this study are significant. Implementing this CNN-based model could significantly improve the accuracy and reliability of IDSs within cloud systems by reducing false positives and effectively identifying diverse cyber threats. It contributes significantly to the theoretical landscape of cybersecurity, showcasing how deep learning can overcome some limitations of traditional IDSs, particularly in dynamic and complex cloud computing settings.

The proposed deep learning-based approach for intrusion detection consists of seven major stages. At first, we acquired a publicly available dataset regarding intrusion detection in the cloud environment. We preprocessed the collected dataset in the second stage to remove missing or null values. Similarly, we performed label encoding and data standardization. We used Pearson correlation matrix analysis in the third stage to select useful features and efficiently train the deep learning models. After that, we used the SOMTE oversampling strategy and down-sampling to obtain a balanced dataset and then split the data into training and testing sets. We trained and tested deep learning models for classifying cyberattacks in the fifth and sixth stages. Finally, we compared the performance of trained deep learning models and proposed the best-performing model for efficient intrusion detection in a cloud environment.

Future works can take several directions to further enhance the effectiveness and applicability of deep learning models in cloud computing cybersecurity.

These include the following:

- Testing the CNN model in a live cloud environment can help assess its performance in real-time scenarios and understand the practical challenges and effectiveness of the model under operational conditions.
- Exploring more optimization strategies, such as genetic algorithms, could improve the efficiency and accuracy of the CNN model.
- Investigating the model's resilience against adversarial attacks designed to deceive machine learning models can lead to more robust cybersecurity defenses.
- Exploring the integration of CNNs with other neural network architectures, such as recurrent neural networks or autoencoders, can yield better detection rates and accuracy, especially for complex attack vectors requiring sequential data analysis or unsupervised learning.

Author Contributions: For this research article, both authors contributed equally to all sections. Conceptualization, W.H.A. and S.S.A.; methodology, W.H.A.; software, W.H.A.; validation, W.H.A. and S.S.A.; formal analysis, S.S.A. and W.H.A.; writing—original draft preparation, review and editing, W.H.A. and S.S.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Taif University, Saudia Arabia, project no. (TU-DSPP-2024-52).

Institutional Review Board Statement: Not applicable.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Acknowledgments: The authors extend their appreciation to Taif University, Saudia Arabia, for supporting this work through project number (TU-DSPP-2024-52).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Jouini, M.; Rabai, L.B.A. A security framework for secure cloud computing environments. In *Cloud security: Concepts, Methodologies, Tools, and Applications*; IGI Global: Hershey, PA, USA, 2019; pp. 249–263.
2. Saini, P.S.; Behal, S.; Bhatia, S. Detection of DDoS attacks using machine learning algorithms. In Proceedings of the 2020 7th International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 12–14 March 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 16–21.
3. Wang, L.; Von Laszewski, G.; Yo unge, A.; He, X.; Kunze, M.; Tao, J.; Fu, C. Cloud computing: A perspective study. *New Gener. Comput.* **2010**, *28*, 137–146. [CrossRef]
4. Bakro, M.; Bisoy, S.K.; Patel, A.K.; Naal, M.A. Performance analysis of cloud computing encryption algorithms. In *Advances in Intelligent Computing and Communication, Proceedings of the ICAC 2020, Colombo, Sri Lanka, 10–11 December 2020*; Springer: Singapore, 2021; pp. 357–367.
5. El Alloussi, H.; Fetjah, L.; Sekkaki, A. L'état de l'art de la sécurité dans le Cloud Computing. In Proceedings of the INTIS 2012, Mohammadia, Morocco, 23–24 November 2012; Volume 3.
6. Gu, J.; Wang, L.; Wang, H.; Wang, S. A novel approach to intrusion detection using SVM ensemble with feature augmentation. *Comput. Secur.* **2019**, *86*, 53–62. [CrossRef]
7. Edeh, D.I. Network Intrusion Detection System Using Deep Learning Technique. Master's Thesis, Department of Computing, University of Turku, Turku, Finland, 2021.
8. Attou, H.; Guezzaz, A.; Benkirane, S.; Azrour, M.; Farhaoui, Y. Cloud-Based Intrusion Detection Approach Using Machine Learning Techniques. *Big Data Min. Anal.* **2023**, *6*, 311–320. [CrossRef]
9. Jyothsna, V.; Manisha, C.; NanduSri, B.S. Intrusion Detection System for Detection of DDoS Attacks in Cloud Environment. *Res. Sq.* 2023. [CrossRef]
10. Aldallal, A. Toward efficient intrusion detection system using hybrid deep learning approach. *Symmetry* **2022**, *14*, 1916. [CrossRef]
11. Srilatha, D.; Shyam, G.K. Cloud-based intrusion detection using kernel fuzzy clustering and optimal type-2 fuzzy neural network. *Clust. Comput.* **2021**, *24*, 2657–2672. [CrossRef]
12. Wu, P. Deep learning for network intrusion detection: Attack recognition with computational intelligence. Master's Thesis, University of New South Wales, Sydney, NSW, Australia, 2020.
13. Mighan, S.N.; Kahani, M. A novel scalable intrusion detection system based on deep learning. *Int. J. Inf. Secur.* **2021**, *20*, 387–403. [CrossRef]
14. Liu, H.; Lang, B. Machine learning and deep learning methods for intrusion detection systems: A survey. *Appl. Sci.* **2019**, *9*, 4396. [CrossRef]

15. Farhan, R.I.; Maolood, A.T.; Hassan, N.F. Optimized deep learning with binary PSO for intrusion detection on CSE-CIC-IDS2018 dataset. *J. Al-Qadisiyah Comput. Sci. Math.* **2020**, *12*, 16–27. [CrossRef]
16. Bamasag, O.; Alsaeedi, A.; Munshi, A.; Alghazzawi, D.; Alshehri, S.; Jamjoom, A. Real-time DDoS flood attack monitoring and detection (RT-AMD) model for cloud computing. *PeerJ Comput. Sci.* **2022**, *7*, e814. [CrossRef]
17. Bhardwaj, A.; Mangat, V.; Vig, R. Hyperband tuned deep neural network with well posed stacked sparse autoencoder for detection of DDoS attacks in cloud. *IEEE Access* **2020**, *8*, 181916–181929. [CrossRef]
18. Khraisat, A.; Gondal, I.; Vamplew, P.; Kamruzzaman, J.; Alazab, A. Hybrid intrusion detection system based on the stacking ensemble of c5 decision tree classifier and one class support vector machine. *Electronics* **2020**, *9*, 173. [CrossRef]
19. Qazi, E.U.H.; Faheem, M.H.; Zia, T. HDLNIDS: Hybrid Deep-Learning-Based Network Intrusion Detection System. *Appl. Sci.* **2023**, *13*, 4921. [CrossRef]
20. Issa, A.S.A.; Albayrak, Z. DDos attack intrusion detection system based on hybridization of CNN and LSTM. *Acta Polytech. Hung.* **2023**, *20*, 1–9.
21. Yin, C.; Zhu, Y.; Fei, J.; He, X. A deep learning approach for intrusion detection using recurrent neural networks. *IEEE Access* **2017**, *5*, 21954–21961. [CrossRef]
22. Chen, L.; Kuang, X.; Xu, A.; Suo, S.; Yang, Y. A novel network intrusion detection system based on CNN. In Proceedings of the 2020 Eighth International Conference on Advanced Cloud and Big Data (CBD), Taiyuan, China, 5–6 December 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 243–247.
23. Nayyar, S.; Arora, S.; Singh, M. Recurrent neural network-based intrusion detection system. In Proceedings of the 2020 International Conference on Communication and Signal Processing (ICCSP), Chennai, India, 28–30 July 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 0136–0140.
24. Farahnakian, F.; Heikkonen, J. A deep auto-encoder based approach for intrusion detection system. In Proceedings of the 2018 20th International Conference on Advanced Communication Technology (ICACT), Chuncheon, Republic of Korea, 11–14 February 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 178–183.
25. Bagyalakshmi, C.; Samundeeswari, E.S. DDoS attack classification on cloud environment using machine learning techniques with different feature selection methods. *Int. J. Adv. Trends Comput. Sci. Eng.* **2020**, *9*, 7301–7308.
26. Naseer, S.; Saleem, Y.; Khalid, S.; Bashir, M.K.; Han, J.; Iqbal, M.M.; Han, K. Enhanced network anomaly detection based on deep neural networks. *IEEE Access* **2018**, *6*, 48231–48246. [CrossRef]
27. Krishna, A.; Lal, A.; Mathewkutty, A.J.; Jacob, D.S.; Hari, M. Intrusion detection and prevention system using deep learning. In Proceedings of the 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC), Coimbatore, India, 2–4 July 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 273–278.
28. Gao, X.; Shan, C.; Hu, C.; Niu, Z.; Liu, Z. An adaptive ensemble machine learning model for intrusion detection. *IEEE Access* **2019**, *7*, 82512–82521. [CrossRef]
29. Ren, W.; Jin, N.; OuYang, L. Phase Space Graph Convolutional Network for Chaotic Time Series Learning. *IEEE Trans. Ind. Inform.* **2024**, *20*, 7576–7584. [CrossRef]
30. IDS 2018 | Datasets | Research | Canadian Institute for Cybersecurity | UNB. (n.d.). Available online: https://www.unb.ca/cic/datasets/ids-2018.html (accessed on 6 December 2023).
31. Shelke, M.S.; Deshmukh, P.R.; Shandilya, V.K. A review on imbalanced data handling using undersampling and oversampling technique. *Int. J. Recent Trends Eng. Res.* **2017**, *3*, 444–449.
32. Jaw, E.; Wang, X. Feature selection and ensemble-based intrusion detection system: An efficient and comprehensive approach. *Symmetry* **2021**, *13*, 1764. [CrossRef]
33. Farhan, R.I.; Maolood, A.T.; Hassan, N.F. Performance analysis of flow-based attacks detection on CSE-CIC-IDS2018 dataset using deep learning. *Indones. J. Electr. Eng. Comput. Sci.* **2020**, *20*, 1413–1418. [CrossRef]
34. Kunang, Y.N.; Nurmaini, S.; Stiawan, D.; Suprapto, B.Y. Attack classification of an intrusion detection system using deep learning and hyperparameter optimization. *J. Inf. Secur. Appl.* **2021**, *58*, 102804. [CrossRef]
35. Alzughaibi, S.; El Khediri, S. A cloud intrusion detection system based on dnn using backpropagation and pso on the cse-cic-ids2018 dataset. *Appl. Sci.* **2023**, *13*, 2276. [CrossRef]
36. Al-Fawa'Reh, M.; Al-Fayoumi, M.; Nashwan, S.; Fraihat, S. Cyber threat intelligence using PCA-DNN model to detect abnormal network behavior. *Egypt. Inform. J.* **2022**, *23*, 173–185. [CrossRef]
37. Hagar, A.; Gawali, B.W. Deep Learning for Improving Attack Detection System Using CSE-CICIDS2018. *NeuroQuantology* **2022**, *20*, 3064.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.