Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 232 (2024) 1636–1645

Procedia Computer Science

www.elsevier.com/locate/procedia

5th International Conference on Industry 4.0 and Smart Manufacturing

Deep learning-based network anomaly detection and classification in an imbalanced cloud environment

Amol D. Vibhute[a,*], Vikram Nakum[a]

[a]*Symbiosis Institute of Computer Studies and Research (SICSR), Symbiosis International (Deemed University), Pune-411016, MH, India.*

Abstract

With the advancements in computer networking, communication between end-to-end systems has increased drastically. However, security issues have also been raised. Thus, detecting anomalies from a complex cloud environment is still challenging. Therefore, the present article proposes the deep Convolutional Neural Network (CNN) model for detecting and classifying near-real-time network intrusions from an imbalanced cloud environment. The random forest model is also offered and implemented to select the best suitable features as input to the CNN model. The experiments were carried out on CSE-CIC-IDS2018 datasets. The results show that the proposed CNN model achieved 97.07% testing accuracy with a 2.93% error rate. The performance of the proposed model was also measured using precision, recall, and f1-score with 98.11, 96.93, and 97.52%. The results are more accurate, precise, promising, and able to detect network anomalies with the highest accuracy and can be successfully used in real-time Industry 4.0 systems.

Keywords: Deep convolutional neural network; network intrusion detection; random forest; CSE-CICIDS2018 datasets; feature selection.

1. Introduction

In recent years, cloud computing has revolutionized how organizations manipulate and produce their IT services [1, 2]. Relating applications and data to complicated cloud conditions has benefits such as scalability, flexibility, and cost efficiency [3, 4, 5]. Nevertheless, this transformation has also presented unknown challenges, especially in network security and intrusion detection [2, 5]. Additionally, the demand for robust Network Intrusion Detection Systems (NIDS) has become crucial [5] due to increasing reliance on computer networks and the growing complexity of cyber threats [6, 7]. Network intrusion detection is paramount in protecting computer networks by identifying and responding to unauthorized access attempts, malicious movements, and possible security breaches [4, 6, 8].

* Corresponding author.
E-mail address: amolvibhute2011@gmail.com

The traditional network intrusion detection methods are less effective because of the inherent complexity of cloud circumstances, with their distributed and dynamic nature [9]. Conventional NIDS methods have frequently relied on rule-based or signature-based methods, which are limited in detecting novel and unknown attacks [8]. The various range of services, virtualized aids, and multi-tenant architectures in cloud environments make a frequently growing attack surface, making it more challenging to detect and react to network intrusions on time [1, 10]. Conversely, the data variations, real-time requirements, heterogeneous and noisy environments, and the fusion of operational and information technology cause several issues and challenges in detecting the network anomalies in Industry 4.0. Moreover, intrusion detection is still challenging due to the heavy network traffic and the enormous velocity and volume of the cloud data. An efficient and scalable detection mechanism is needed in a cloud network with an immense scale and increased data rates capable of processing and analyzing large amounts of network data in real time [1, 10].

Recently, advanced machine learning [7, 11] and artificial intelligence [12] methods have been widely used in Industry 4.0 to address these challenges for detecting network intrusion in complex cloud environments [3]. Furthermore, deep learning techniques, especially deep CNNs and Recurrent Neural Networks (RNNs) have arisen as promising solutions for enhancing the accuracy and efficiency of network intrusion detection [11, 13].

This paper delves into applying deep CNNs for network intrusion detection. We investigate the architecture and design deliberations of deep CNN-based NIDS models. The network intrusion detection systems can acquire increased accuracy, quicker response times, and enhanced strength against emerging threats via deep learning models. Computer network security can be improved by merging deep learning methods with NIDS, allowing users to better safeguard their valuable data and systems from unauthorized access and malicious activities.

The significant contributions of the present research are: (1) we renovate the raw data, which confirms the quality of data through the elimination of unwanted values from the dataset that is free from any inconsistencies, (2) we develop a robust random forest model to select the unique features from complex data. This approach combines multiple decision trees to classify the features and reduce the dimensionality of the data since intrusion detection is a non-linear issue, (3) we propose the deep CNN model to identify and classify the network attacks. The model uses multiple layers and hyperparameters to train it, which enhances the performances of the developed model, and (4) we evaluate the efficiency of our proposed deep learning model using near real-time datasets and compare the results with recent literature.

The remainder of the paper consists of related work in section two. The datasets used and the proposed methodology are presented in section three, which includes the data pre-processing, feature selection, and model development. Section four discusses the results and their accuracies obtained using the proposed deep learning model. The last section concludes the present study.

2. Related work

Several researchers have researched detecting network intrusion with machine and deep learning techniques with some limits. For instance, the study [14] implemented deep learning models such as CNN and Long Short-Term Memory (LSTM) with Apache spark model tested on the CSE-CICIDS2018 dataset with 97.8%, 97.7%, and 99.9% accuracies, respectively. However, the training time of the CNN and LSTM models is 150.26 and 125.29 minutes, respectively. Similarly, the research [15] on network intrusion using deep hybrid learning has been done on CSE-CICIDS2018 and DARPA-IDS datasets with only 66.38, 63.69, 58.52, 63.34, 94.61, and 97.11 percentage accuracies for LSTM-SGDM, LSTM-ADAM, CNN, CNN-LSTM, RC-NN, and DKNN models. It is observed from their research [15] that the correctness of the CNN model is significantly less, i.e., 58.52%, only that could be enhanced using the best features selected via ensemble learning methods. The research [16] on KDD and CSE-CICIDS2018 datasets used CNN and RNN models. However, they created the RGB images of the used datasets and evaluated the models with the highest accuracy. The deep learning model was implemented with only 83.87% accuracy using NSL-KDD datasets [16, 17]. Likewise, the NSL-KDD dataset was also utilized to build the deep learning model with 90.73% accuracy [7]. Moreover, the detection accuracy was only 81.87% with the deep neural network model on NSL-KDD datasets for intrusion detection [9]. Furthermore, deep neural network and LSTM models were used in [18] on the UNSW-NB15 dataset with 95.05 and 88.75 accuracies, which are very few for binary classification. In addition, the study [8] conducted on the UNSW-NB15 dataset using a deep learning-based CNN model has achieved 95.6% accuracy for only 25% testing dataset to detect the ten network attacks.

These earlier researchers either obtained low accuracy or used simple datasets. Additionally, the previous studies mainly focused on binary classification via complex datasets such as CSE-CICIDS2018 or UNSW-NB15. However, deep learning-based models were also successfully implemented with more than 95% accuracy on simple datasets like NSL-KDD data. The CSE-CICIDS2018 datasets are complex, and their number of features is higher than others, which is still challenging to detect and implement. The accuracy obtained via deep learning models on this data still needs to be improved, which motivates us to work on such research.

3. Materials and Methods

3.1. The Datasets

The CSE-CICIDS2018 (Canadian Institute for Cybersecurity - Intrusion Detection System 2018) dataset was developed by the CIC at the University of New Brunswick and is a widely used dataset in the field of cybersecurity for intrusion detection and network analysis research [14, 15, 16]. It consists of various network traffic captures, including benign and malicious activities, allowing researchers and practitioners to develop and evaluate intrusion detection algorithms and systems. It covers a wide range of network attack scenarios and includes labelled data for different types of attacks. In the present study, we used four days of datasets such as ”Wednesday-14-02-2018 and 21-02-2018”, ”Thursday-15-02-2018”, and ”Friday-16-02-2018” collected using the CICFlowMeter. The dataset is nearly 1.30 GB and contains four CSV files with 5,43,589 entries and 80 features in each file [19].

3.2. Data Pre-processing

The raw data were pre-processed to remove unwanted or duplicate values. In addition, the null or, missing or zero data was cleaned and corrected to eliminate any errors in data. Finally, the data was prepared for further processing. Some missing values were also removed from the data. Latterly, the dataset is converted into the float data format for further processing [14] while splitting it into input features (x) and target variables (y). The final pre-processed dataset was numeric and free from any inconsistencies.

3.3. Feature selection using Random Forest

The random forest algorithm [5, 20] is an ensemble learning method that fuses several decision trees to classify the features [3]. It is widely used for classification and feature selection tasks due to its effectiveness and versatility. The random forest measures feature importance based on the information gain or reduction in impurity accomplished by splitting on a particular feature. This information can be used for feature selection [20].

In sci-kit-learn, the RandomForestClassifier is used to learn and select the essential features. The dataset separations are 70% for training and 30% for testing with n_estimators=1000 (decision trees) and random_state=40. The random forest method took 653.2871 seconds to train the model. Subsequently, the essential features were selected, shown in Figure 2 from the random forest classifier model based on the constant threshold values of 0.015. The random forest selector model provided seventeen essential features.

3.4. Data Preparation

The selected data was prepared with three separate datasets for three feature labels (shaping the data for CNN), removed the bias from data, and distributed in equal formation with n_samples=3000 and random_state=123 for all the data.

3.5. Deep Convolutional Neural Network

The deep CNN model is the widely popular form of neural network that analyzes the vast amounts of network traffic data generated in cloud environments and works superior on complex features [11, 14]. By leveraging the deep representations learned from training data, these models can detect and classify network attacks with higher precision and lower false-positive rates [9, 11, 14]. The crucial advantage of deep CNNs lies in their capability to automatically learn features directly from raw network traffic data, eliminating the need for handcrafted features [11]. Furthermore, deep CNNs can adapt and generalize well to new and previously unseen attack types, making them suitable for detecting zero-day attacks [8, 9].

The CNN architecture consists of several layers and the activation functions to activate the model. The CNN model's major hyperparameters are the input layer, 1D convolutional layers, max-pooling layers, full connection layers, and output layers with its feedforward neural network approach. In the present study, we used four different datasets and CNN's sequential_44 model with a 1D convolutional layer with 128 filters, increasing kernel sizes and ReLU activation function. The 1D sequential models were developed with filters=64, kernel_size=3, MaxPooling1D (pool_size=2, strides=2, padding='same'), and flattened layers. The fully connected layer with 128 units and ReLU activation function was run with Dense-units=64, activation='SoftMax'. In addition, the dropout layer was set to 0.5 to prevent overfitting and the output layer was added with the SoftMax activation function. Lastly, the model was compiled with Adam optimizer and categorical cross-entropy loss function.

4. Results and discussion

The personal Windows 10 Home 64-bit operating systems, Intel(R) Core (TM) i5-7200U, and Graphics-2.71 GHz with 8 GB RAM were used to implement and test the machine and deep learning models. The present study's proposed approach was computed using the Python 3.10 version in Anaconda Jupyter Notebook—the Python libraries include numpy, pandas, matplotlib, sklearn, Keras, TensorFlow, etc. Initially, the original raw data was pre-processed to check for the null values, replace the null and infinite values and the count frequency of each attack type. In the original data, there were five attacks: FTP-BruteForce, DoS attacks-SlowHTTPTest, DDOS attack-HOIC, DoS attacks-GoldenEye, and DoS attacks-Slowloris and one normal: Benign class. The frequency of each attack type is shown in Figure 1.

Subsequently, the essential seventeen features were selected from eighty features using the random forest algorithm [20], as depicted in Figure 2. Hence, the unwanted features were eliminated from further processing without losing important information. The resultant dataset had only seventeen most noteworthy features and 5,43,589 rows after the feature selection using the random forest method [20]. Figure 2 demonstrates the critical features selected via the random forest.

The data split into train and test concerning seventy and thirty percent, respectively, and models were trained and tested accordingly. The proposed CNN model was implemented with their hyperparameters such as hidden nodes' 192', batch size '64', and learning_rate=0.0001, 'categorical cross-entropy loss, activation function= 'SoftMax', with 'Adam' optimizer. The total trainable and non-trainable parameters were '30,214', '29,830', and '384', respectively. Thus, the training and testing time of the CNN model was '20.66' minutes and '0.21' seconds, respectively. The obtained training accuracy, training loss, validation accuracy and loss were correspondingly 0.9880, 0.0531, 0.9707, and 0.3591. Table 1 shows the implemented CNN model's training and testing accuracy and loss.

The performance of the CNN model was evaluated using an error matrix, 10-fold cross-validation, and metrics like accuracy, precision, recall, and the F1-score executed using Eq. s (1), (2), (3), and (4), respectively.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$F1 - Sccore = 2 * \frac{Recall * Precision}{Recall + Precision} \tag{4}$$

Where TP, TN, FP, and FN are truly positive, predict one, and the actual is also one. True negative predicts zero, and actual is also zero. A false positive indicates one, but the essential is zero, and a false negative predicts zero but is one consistently [3, 10].

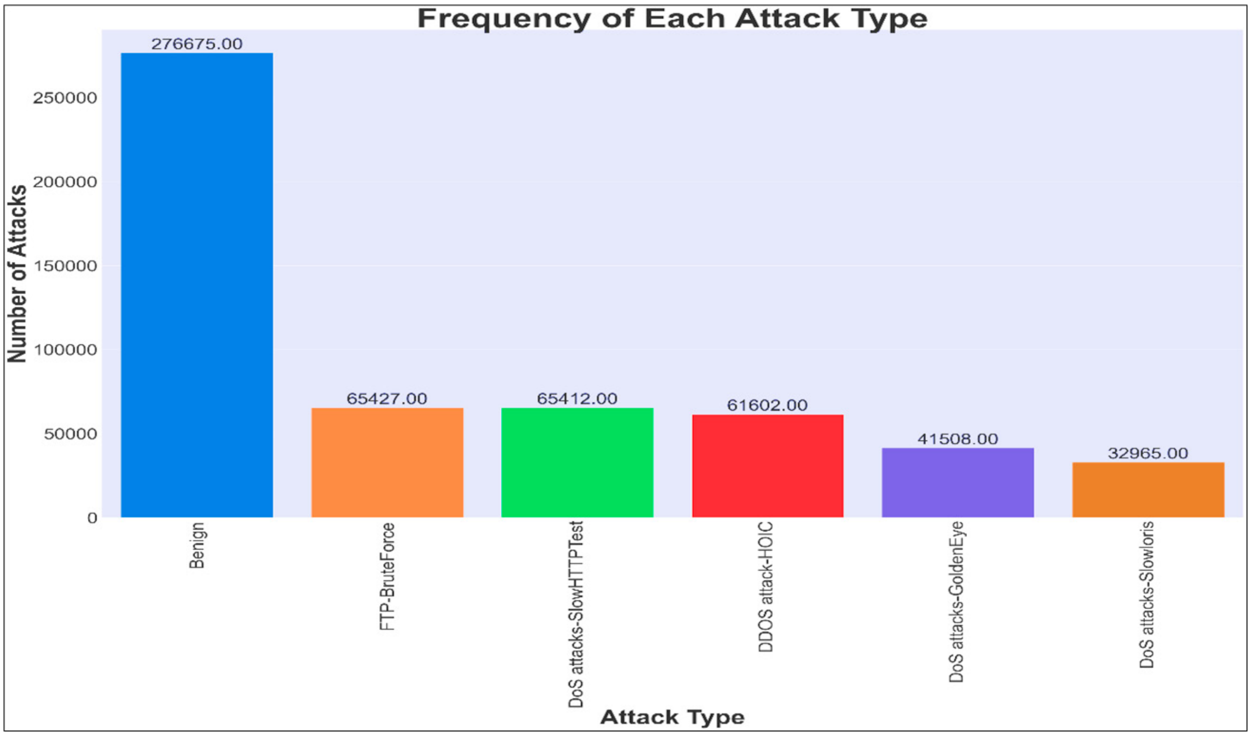


Fig. 1. The frequency count of each attack type.

Table 1. Training and testing accuracy and loss of CNN model.

CNN	Training (%)	Validation (%)
Accuracy	98.80	97.07
Loss	0.531	3.591

The training and testing results are revealed in Table 2, and Figure 3 shows that the CNN model has provided the highest training and testing accuracy, 98.80% and 97.07%. The precision, recall, and F1 scores on training datasets were 99.89%, 97.83%, and 98.94%, respectively, and testing datasets were 98.11%, 96.93%, and 97.51%, respectively. However, the highest precision was achieved for both the training and testing datasets (Table 2 and Figure 3).

Table 2. The CNN model results on training and testing datasets.

Performance metrics	Training results (%)	Testing results (%)
Accuracy	98.80	97.07
Precision	99.89	98.11
Recall	97.83	96.93
F1-Score	98.94	97.51

The class-specific performances on testing datasets concerning the precision, recall and F1-values of the computed CNN model are depicted in Table 3. The CNN model's accuracy and loss were executed for every epoch's training and testing sets. In our case, we used forty epochs to correctly train the CNN model due to the complexity of the data for showing the train and test accuracy and loss. The accuracy plot has demonstrated that the CNN model has been successfully trained from 13 number epochs with more than 98% accuracy. In contrast, the testing accuracy was highest on 20, 34, and 40 epochs (Figure 4). Thus, the model has been accurately trained to classify the numerous

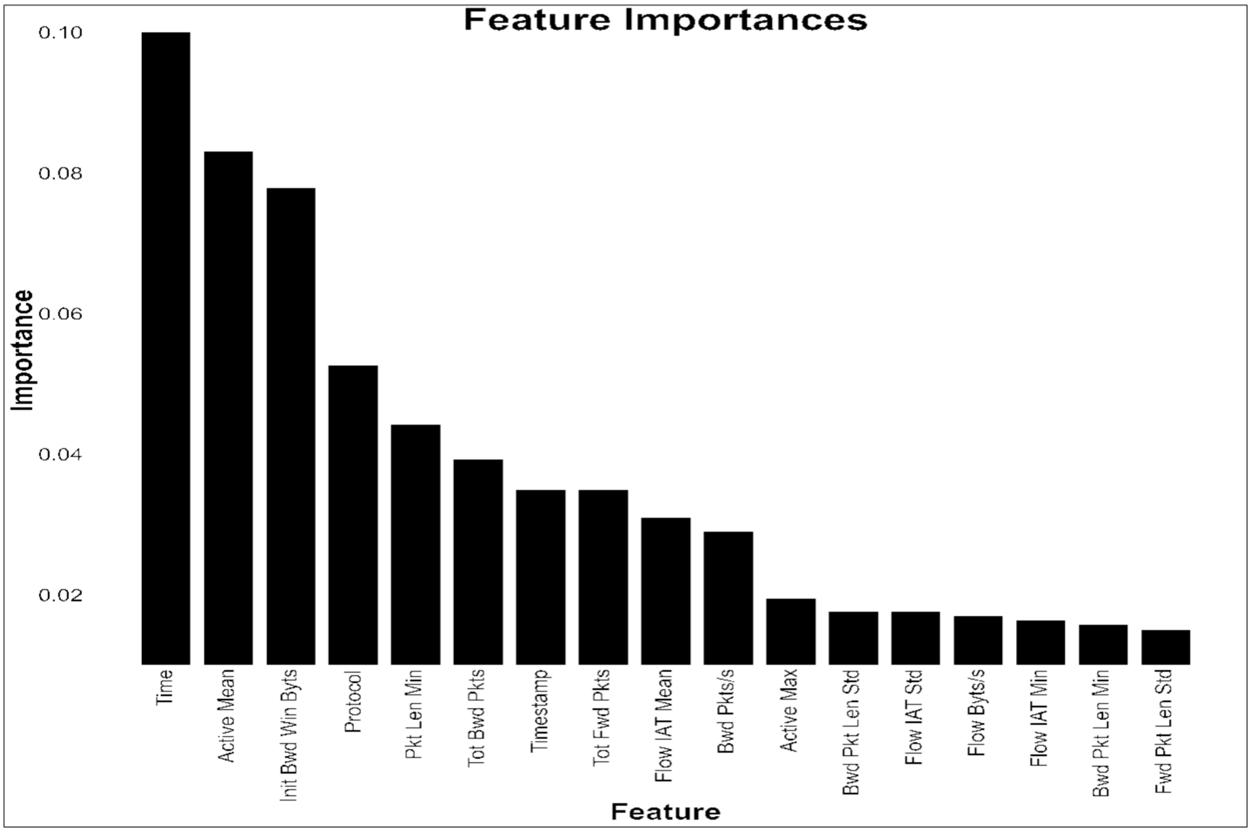


Fig. 2. The selected seventeen important features by random forest method.

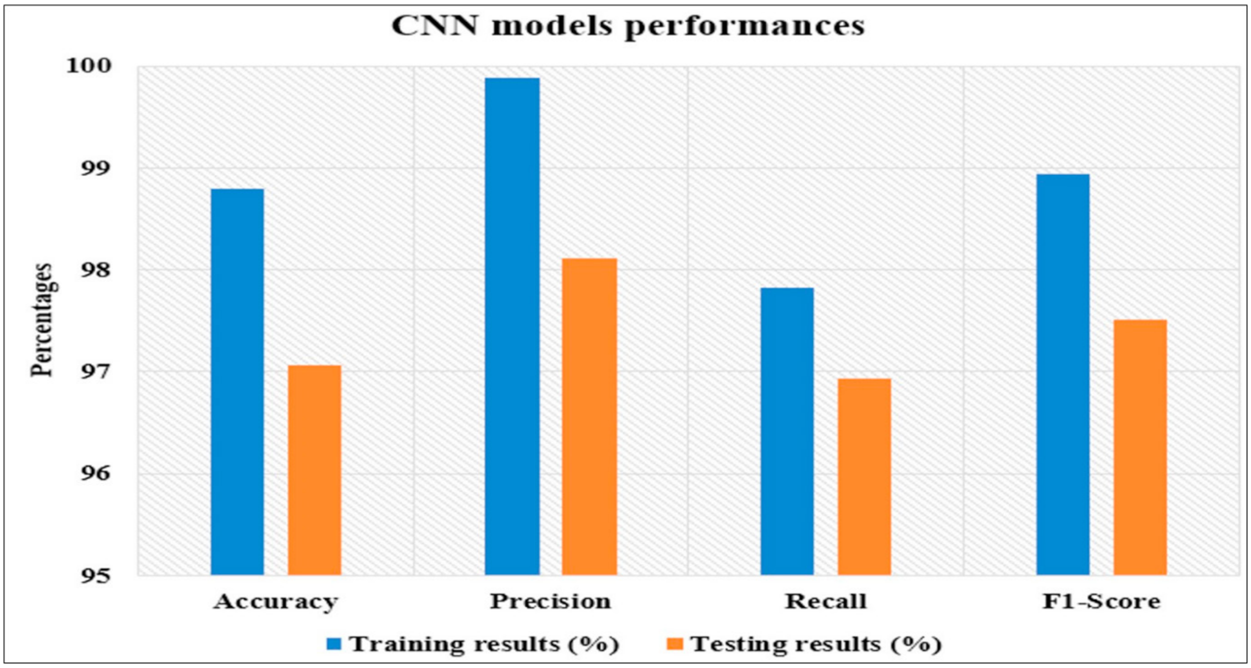


Fig. 3. CNN model performances.

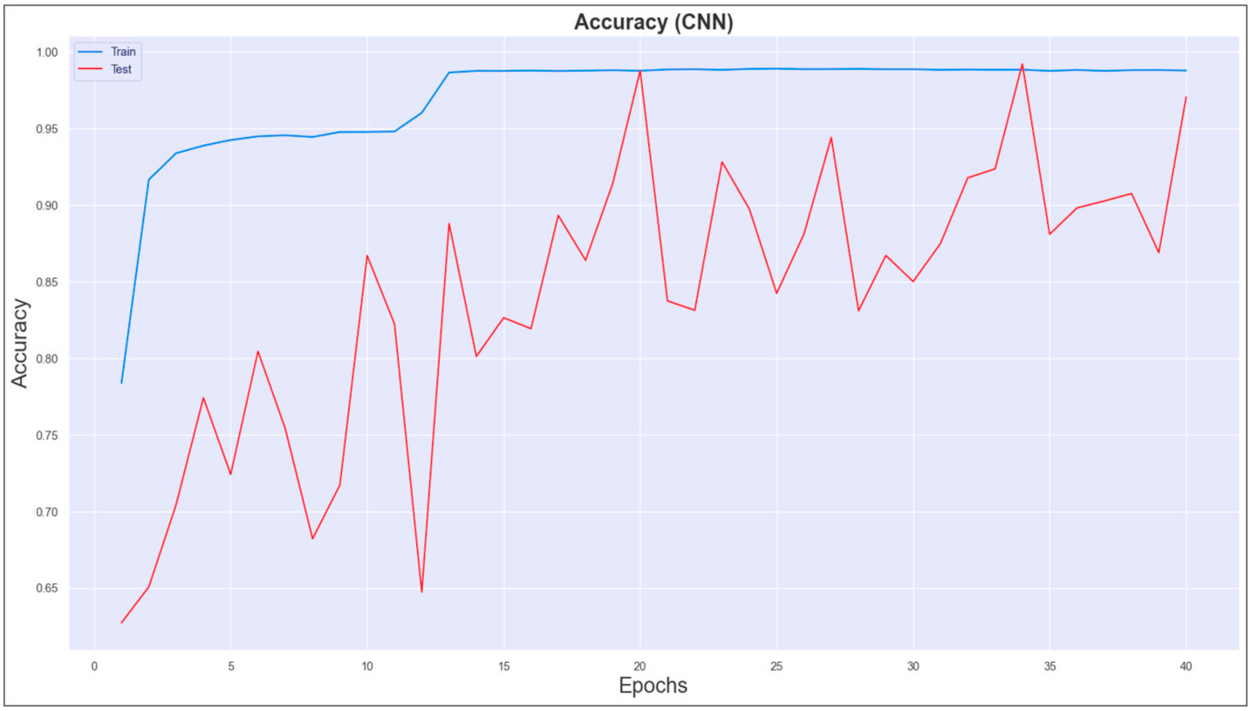


Fig. 4. The CNN models accuracy plot for forty epochs for training and testing data.

attacks. Alternatively, the loss shows the total number of errors for every training sample. In the present study, we achieved the highest training and testing accuracy and the lowest loss (Figures 4 and 5). Figure 4 demonstrates the training and testing accuracy of the CNN model with forty epochs. The training loss was only 2%, also constant from 13 epochs. The testing loss was low at 20, 34, and 40 epochs (Figure 5). The loss results show that the erroneous classification is significantly less with forty epochs. Figure 5 illustrates the loss curve over the successive epochs throughout the training and testing stages. The proposed CNN model has produced accuracy for all the multiple classes with few training and testing times.

Table 3. Classspecific performances of the CNN model on testing datasets.

Class	Precision (%)	Recall	F1score (%)
Benign	0.93	0.92	0.93
DDOS attackHOIC	1.00	0.92	0.96
DoS attacksGoldenEye	0.98	1.00	0.99
DoS attacksSlowHTTPTest	1.00	1.00	1.00
DoS attacksSlowloris	0.92	0.99	0.95
FTPBruteForce	1.00	0.99	0.99
Test: accuracy = 0.97 and loss = 0.35			

It is observed from the loss curve that the testing set has very few inaccurate estimates. Thus, the CNN model has correctly trained with forty epochs. The robustness of the CNN model has been evaluated through the error matrix (Figure 6) generated from the CNN model, which shows that the correctly classified instances are satisfactory. Figure 6 shows the error matrix generated via the CNN model using the chosen datasets. The error matrix is used for solving classification issues because it precisely signifies real and estimated classes. In our experiment, only 2.94% of samples were wrongly classified with other classes, which shows the CNN model has resulted accurately. The expected accuracy of the implemented CNN model was 97.06%, the highest in detecting and classifying the correct

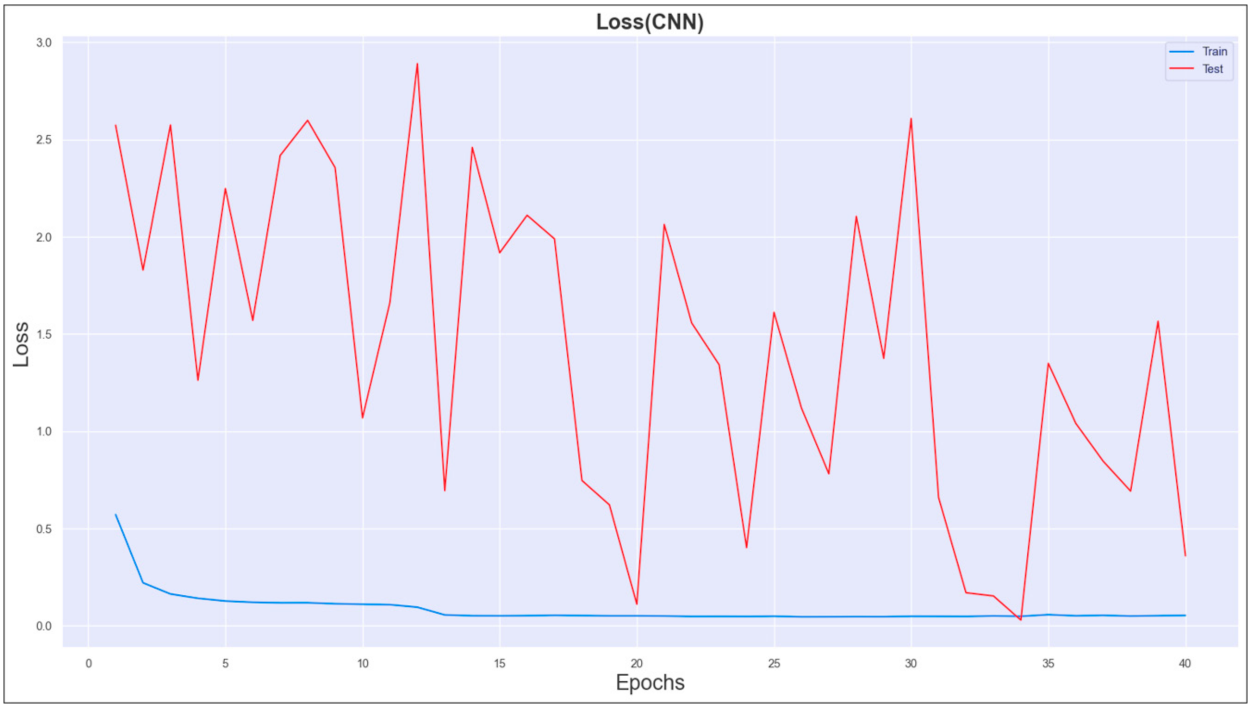


Fig. 5. The CNN model's loss plot for forty epochs for training and testing data.

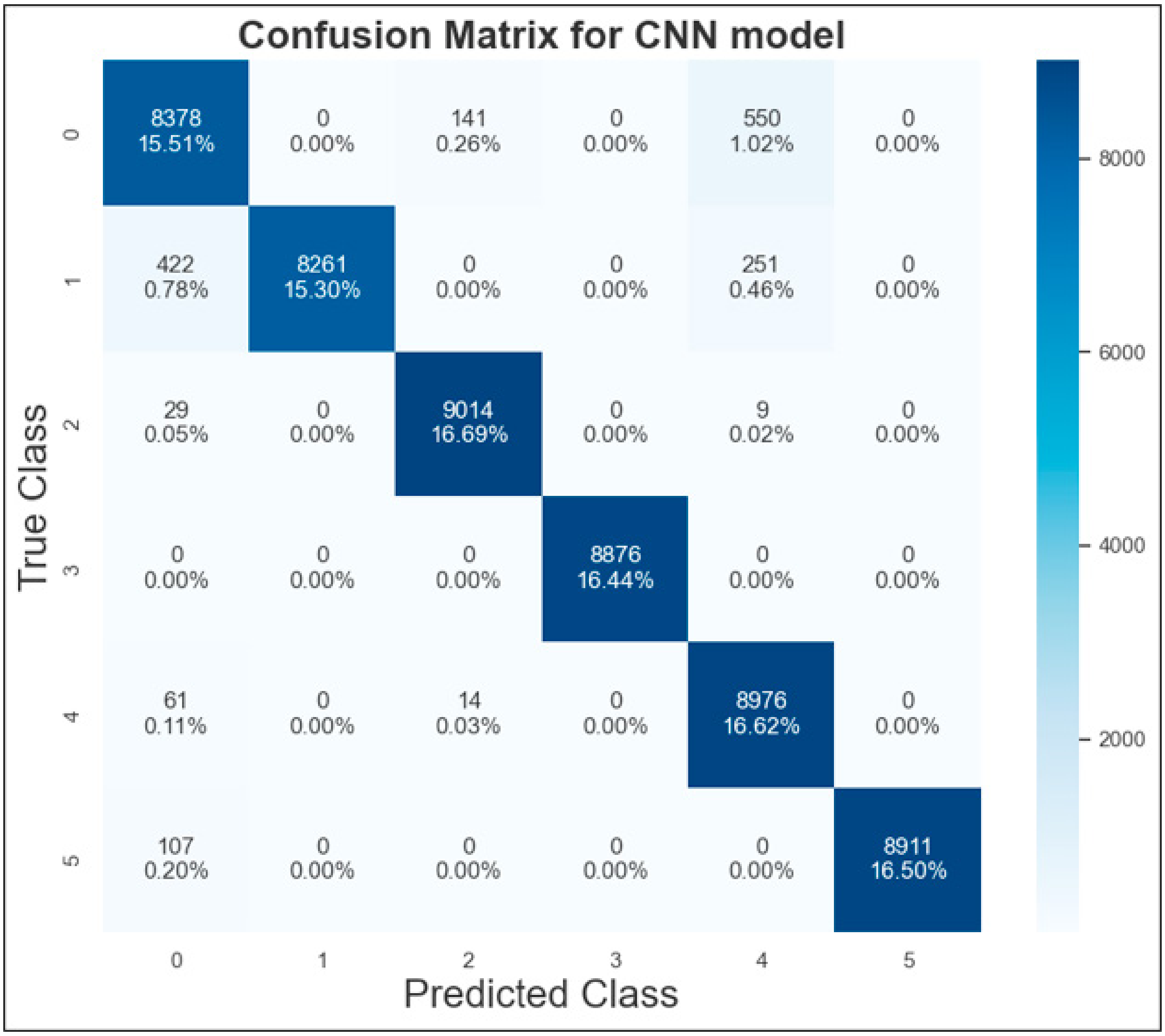
multiple classes of attacks from complex datasets. Thus, it is confirmed that the proposed implemented CNN model is superior to earlier research [7, 8, 9, 15, 17, 18] for detecting and classifying the network attacks from complex datasets with the highest accuracy.

5. Conclusions

The present study contributes towards renovating the raw data, selecting the best features from complex data using the random forest method, and developing the CNN model to successfully identify and classify the network attacks. The random forest method has provided the best features for creating the deep learning-based CNN model. The proposed CNN model accurately detects and classifies network attacks from complicated datasets. The CNN model took only '20.66' minutes and '0.21' seconds to train and test the datasets, respectively. The experimental results show that the CNN model has given 98.80 and 97.07% accuracy on training and testing datasets with only forty epochs. It is concluded that the dimensionality of the data can be reduced, and the best features can be selected using the random forest method with the highest success rate and without losing essential information. Moreover, the deep CNN model is superior for complex datasets with the highest detection rate. The outcome of the present research can be used in real-time monitoring and response, allowing industries to detect and respond to cyber threats and attacks as they happen rather than the facts. Furthermore, the present study's outcome is essential in Industry 4.0 to provide real-time insights into network behaviour, promptly make data-driven decisions, respond to emerging issues, and adapt to changing market conditions more effectively. In addition, the present study can also be crucial for mitigating and protecting digital information in driving the evolving landscape of Industry 4.0. In the future, the accuracy of the proposed CNN model can be enhanced up to 100%.

Acknowledgements

The authors would like to thank the data provider for conducting this research.



True \ Predicted	0	1	2	3	4	5
0	8378 15.51%	0 0.00%	141 0.26%	0 0.00%	550 1.02%	0 0.00%
1	422 0.78%	8261 15.30%	0 0.00%	0 0.00%	251 0.46%	0 0.00%
2	29 0.05%	0 0.00%	9014 16.69%	0 0.00%	9 0.02%	0 0.00%
3	0 0.00%	0 0.00%	0 0.00%	8876 16.44%	0 0.00%	0 0.00%
4	61 0.11%	0 0.00%	14 0.03%	0 0.00%	8976 16.62%	0 0.00%
5	107 0.20%	0 0.00%	0 0.00%	0 0.00%	0 0.00%	8911 16.50%

Fig. 6. CNN modal derived confusion matrix implemented on the testing dataset.

References

[1] Jaber, A. N., and Rehman, S. U. (2020). "FCM–SVM based intrusion detection system for cloud computing environment." *Cluster Computing* **23** : 3221-3231.

[2] Atefinia, R., and Ahmadi, M. (2021). "Network intrusion detection using multi-architectural modular deep neural network." *The Journal of Supercomputing* **77** : 3571-3593.

[3] Krishnaveni, S., Sivamohan, S., Sridhar, S. S., and Prabakaran, S. (2021). "Efficient feature selection and classification through ensemble method for network intrusion detection on cloud computing." *Cluster Computing* **23** (3): 1761-1779.

[4] Pooja, T. S., and Shrinivasacharya, P. (2021). "Evaluating neural networks using Bi-Directional LSTM for network IDS (intrusion detection systems) in cyber security." *Global Transitions Proceedings* **2** (2): 448-454.

[5] Shahzad, F., Mannan, A., Javed, A. R., Almadhor, A. S., Baker, T., and Al-Jumeily OBE, D. (2022). "Cloud-based multiclass anomaly detection and categorization using ensemble learning." *Journal of Cloud Computing* **11** (1) : 1-12.

[6] Laghrissi, F., Douzi, S., Douzi, K., and Hssina, B. (2021). "Intrusion detection systems using long short-term memory (LSTM)." *Journal of Big Data* **8** (1) : 65.

[7] Fu, Y., Du, Y., Cao, Z., Li, Q., and Xiang, W. (2022). "A deep learning model for network intrusion detection with imbalanced data." *Electronics* **11** (6): 898.

[8] Ashiku, L., and Dagli, C. (2021). "Network intrusion detection system using deep learning." *Procedia Computer Science* **185** : 239-247.

[9] Thirimanne, S. P., Jayawardana, L., Yasakethu, L., Liyanaarachchi, P., and Hewage, C. (2022). "Deep neural network based real-time intrusion detection system." *SN Computer Science* **3** (2): 145.

[10] Besharati, E., Naderan, M., and Namjoo, E. (2019). "LR-HIDS: logistic regression host-based intrusion detection system for cloud environments." *Journal of Ambient Intelligence and Humanized Computing* **10**: 3669-3692.

[11] Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., Al-Nemrat, A., and Venkatraman, S. (2019). "Deep learning approach for intelligent intrusion detection system." *IEEE Access* **7**: 41525-41550.

[12] Gao, Y., Liu, Y., Jin, Y., Chen, J., and Wu, H. (2018). "A novel semi-supervised learning approach for network intrusion detection on cloud-based robotic system." *IEEE Access* **6**: 50927-50938.

[13] Khan, A. R., Kashif, M., Jhaveri, R. H., Raut, R., Saba, T., and Bahaj, S. A. (2022). "Deep learning for intrusion detection and security of Internet of things (IoT): current analysis, challenges, and possible solutions." *Security and Communication Networks* 2022.

[14] Hagar, A. A., and Gawali, B. W. (2022). "Apache Spark and Deep Learning Models for High-Performance Network Intrusion Detection Using CSE-CIC-IDS2018." *Computational Intelligence and Neuroscience* 2022.

[15] Mayuranathan, M., Saravanan, S. K., Muthusenthil, B., and Samydurai, A. (2022). "An efficient optimal security system for intrusion detection in cloud computing environment using hybrid deep learning technique." *Advances in Engineering Software* **173**: 103236.

[16] Kim, J., Kim, J., Kim, H., Shim, M., and Choi, E. (2020). "CNN-based network intrusion detection against denial-of-service attacks." *Electronics* **9** (6): 916.

[17] Parampottupadam, S., and Moldovann, A. N. (2018, June). "Cloud-based real-time network intrusion detection using deep learning." *In 2018 International Conference on Cyber Security and Protection of Digital Services (Cyber Security), IEEE* : 1-8.

[18] Archana, HP, C., Khushi, Nandini, P., Sivaraman, and Honnavalli, P. (2021, August). "Cloud-based network intrusion detection system using deep learning." *In The 7th Annual International Conference on Arab Women in Computing in Conjunction with the 2nd Forum of Women in Research* : 1-6.

[19] https://www.unb.ca/cic/datasets/ids-2018.html Accessed on 02 May 2023.

[20] Li, X., Chen, W., Zhang, Q., and Wu, L. (2020). "Building auto-encoder intrusion detection system based on random forest feature selection." *Computers Security* **95**: 101851.