JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Network anomaly detection and categorization usingdeep learning in an unbalanced cloud environment

Annu Kumari Singh1, Tushar Giri2

1M. Tech Scholar, Department of CSE, Institute of Technology& Management Lucknow UP India

2Assistant Professor, Department of CSE, GCRG. Group of Institutions, Lucknow, UP, India

Abstract-In this paper, a deep Convolutional Neural Network (CNN) model is presented for the purpose of identifying and categorizing network intrusions in near real time from an unbalanced cloud environment. In order to choose the most appropriate characteristics to feed into the CNN model, the random forest model is also provided and used. The CSE-CIC-IDS2018 datasets were used in the experiments. The outcomes demonstrate that the suggested CNN model has a testing accuracy of 97.07% and an error rate of 2.93%. The accuracy, recall, and f1-score of the suggested model were also evaluated, with results of 98.11, 96.93, and 97.52%, respectively. The outcomes are more exact, accurate, and promising. They can be successfully applied in real-time Industry 4.0 systems and are capable of detecting network anomalies with the

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

1. Introduction

Cloud computing has completely changed how businesses manage and generate their IT services in the last few years [1, 2]. Benefits like scalability, flexibility, and cost effectiveness can be obtained by connecting apps and data to complex cloud environments [3, 4, 5]. However, this change has also brought with it unanticipated difficulties, particularly in intrusion detection and network security [2, 5]. Furthermore, as our reliance on computer networks grows and cyber threats become more sophisticated, the need for reliable Network Intrusion Detection Systems (NIDS) has become essential [5]. In order to safeguard computer networks from malicious activity, unauthorized access attempts, and potential security breaches, network intrusion detection is essential. Because cloud environments are scattered and dynamic by design, they are inherently more complex than traditional network intrusion detection techniques [9]. The use of rule-based or signature-based techniques, which have limitations in identifying new and unidentified threats, has been a common practice in conventional NIDS methodologies [8]. The different A constantly expanding attack surface is created by the variety of services, virtualized tools, and multi-tenant architectures seen in cloud environments, making it more difficult to identify and respond promptly to network intrusions [1, 10]. In contrast, there are a number of problems and difficulties in detecting network abnormalities in Industry 4.0 due to data variances, real-time requirements, heterogeneous and loud settings, and the merging of operational and information technologies. In addition, intrusion detection is difficult because of the massive amount of network traffic and the In a cloud network with enormous scale and higher data rates that can collect and analyze massive volumes of network data in real time, an effective and scalable detection method is required [1, 10].Modern artificial intelligence [12] and machine learning [7, 11] techniques have been extensively employed in These difficulties in identifying network intrusion in intricate cloud settings will be addressed by Industry 4.0 [3]. Moreover, recurrent neural networks (RNNs) and deep CNNs, in particular, have emerged as potential methods for improving the precision and effectiveness of network intrusion detection [11, 13]. The use of deep CNNs for network intrusion detection is explored in this research. We look into the deep CNN-based NIDS models' architecture and design considerations. Through the use of deep learning models, network intrusion detection systems can become more resilient to new threats and have better accuracy and response times. Workstation Combining deep learning techniques with NIDS can enhance network security and help users better protect their systems and important data from malicious activity and illegal access. The current study makes two important contributions: (1) we improve the raw data, which verifies the quality of the data by removing unwanted values from the dataset that is devoid of any inconsistencies; and (2) we create a strong random forest model to identify the distinctive features from complicated data. Since intrusion detection is a non-linear problem, this method uses numerous decision trees to categorize the characteristics and minimize the dimensionality of the data. (3) To identify and classify network attacks, we propose the deep CNN model. The model makes use of We train the developed model with several layers and hyperparameters, which improves its performance. (4) We use near real-time datasets to assess the effectiveness of our suggested deep learning model, and we compare the findings with recent research. The related work from section two makes up the rest of the paper. Section three presents the datasets that were used, along with the suggested technique that covers feature selection, model construction, and data pre-processing. The results and the accuracy of the results are covered in Section 4.

Related work 2With some limitations, a number of researchers have looked into using machine and deep learning approaches to detect network intrusions. For example, the study [14] used Apache Spark to test deep learning models like CNN and Long Short-Term Memory (LSTM) with 97.8%, 97.7%, and 99.9% success rates on the CSE-CICIDS2018 dataset. accuracy, correspondingly. On the other hand, the CNN and LSTM models require 150.26 and 125.29 minutes of training time, respectively. Comparably, LSTM-SGDM, LSTM-ADAM, CNN, CNN-LSTM, RC-NN, and DKNN models only had percentage accuracies of 66.38, 63.69, 58.52, 63.34, 94.61, and 97.11 in the study [15] on network intrusion using deep hybrid learning conducted on the CSECICIDS2018 and DARPA-IDS datasets. Based on their research [15], it is found that the CNN model's correctness is much lower, at 58.52%. Only that could be improved by utilizing the top characteristics chosen through ensemble learning techniques. RNN and CNN models were employed in the study [16] on the KDD and CSE-CICIDS2018 datasets. Nevertheless, they produced the RGB images of the datasets that were utilized and assessed the models using the maximum precision. Using NSL-KDD datasets, the deep learning model was developed with only 83.87% accuracy [16, 17].

Likewise, a deep learning model with an accuracy of 90.73% was constructed using the NSL-KDD dataset [7]. Furthermore, using the deep neural network model for intrusion detection on NSL-KDD datasets, the detection accuracy was only 81.87% [9]. Additionally, [18] employed LSTM and deep neural network models on the UNSWNB15 dataset, achieving 88.75 and 95.05 accuracies, respectively, which are extremely low for binary classification. Furthermore, a deep learning-based CNN model used in the study [8] on the UNSW-NB15 dataset was able to detect the 10 network attacks with 95.6% accuracy for just 25% of the testing dataset. These past researchers employed basic datasets or produced results with limited accuracy. Furthermore, the majority of the prior research employed complicated datasets as CSE-CICIDS2018 or UNSW-NB15 to study binary classification. On the other hand, deep learning-based models were also effectively applied with over 95% accuracy on basic datasets. similar to NSL-KDD data. The complexity and increased feature count of the CSE-CICIDS2018 datasets make them harder to find and use than other datasets. Our motivation to work on this topic stems from the fact that the accuracy achieved by deep learning models on this data still needs to be enhanced.

3. Materials and Methods

3.1. The Datasets

The University of New Brunswick's CIC created the CSE-CICIDS2018 (Canadian Institute for Cybersecurity - Intrusion Detection System 2018) dataset, which is extensively utilized in the cybersecurity community for network analysis and intrusion detection studies [14, 15, 16]. It is made up of different network traffic captures, such as benign and malevolent actions, enabling scientists and professionals to create and assess systems and algorithms for intrusion detection. It offers labeled data for several attack types and covers a broad range of network attack scenarios. We used four days' worth of statistics from the CICFlowMeter, including "Wednesday, 14-02-2018 and 21-02-2018," "Thursday, 15-02-2018," and "Friday, 16-02-2018," for the current study. The dataset, which has four CSV files with 5,43,589 items and 80 features, is around 1.30 GB in size.

3.2.Data Pre-processing

Unwanted or redundant values were eliminated from the raw data through pre-processing. To remove any inaccuracies in the data, the null, missing, or zero data was also cleaned and repaired. The data was then ready for additional processing.Also, certain missing values were eliminated from the information. The dataset has recently been transformed into the float data format for separating it into input characteristics (x) and target variables (y) for additional processing [14]. The preprocessed dataset that was produced in the end was numerical and error-free.

3.3. *Feature selection using Random Forest*

An ensemble learning technique called the random forest algorithm [5, 20] combines many decision trees to classify the features [3]. Because of its efficiency and adaptability, it is frequently utilized for jobs involving classification and feature selection. The random forest calculates the significance of each feature according to the amount of information gained or the amount of impurity reduced by
dividing based on a specific attribute. Feature selection can be done using this information [20]. The RandomForestClassifier in sci-kit-learn is used to identify and learn the key features. When using n estimators = 1000 (decision trees) and random state = 40, the dataset separations are 70% for training and 30% for testing. It took 653.2871 seconds for the random forest approach to train the model. Then, the crucial elements were chosen, as seen in

3.4. *Data Preparation*

The chosen data was cleaned up by dividing it into three distinct datasets for each of the three feature labels (forming the data for CNN), removing bias, and distributing the data evenly over n samples = 3000 and random state = 123 for all of the data.

3.5. *Deep Convolutional Neural Network*

The most extensively used type of neural network is the deep CNN model, which performs better on complicated features and analyzes enormous volumes of network traffic data produced in cloud environments [11, 14]. These models can identify and categorize network threats more accurately and with fewer false-positive rates by utilizing the deep representations that they have acquired from training data [9, 11, 14]. The primary benefit of deep CNNs is theircapacityto Without the requirement for manually created features, features can be automatically learned from raw network traffic data [11]. Deep CNNs are also good at generalizing and adapting to novel and never-before-seen attack types, which makes them useful for identifying zero-day attacks [8, 9]. In order to activate the model, the CNN architecture is composed of multiple layers and activation functions. The input layer, 1D convolutional layers, max-pooling layers, full connection layers, and output layers with its feedforward neural network technique are the main hyperparameters of the CNN model. In this investigation, we employed four distinct datasets, rising kernel sizes, and the ReLU activation function of CNN's sequential 44 model, which has a 1D convolutional layer with 128 filters. Filters=64, kernel size=3, MaxPooling1D (pool size=2, strides=2, padding='same'), and flattened layers were used in the development of the 1D sequential models. Dense-units=64, activation='SoftMax' was used to operate the fully linked layer with 128 units and the ReLU activation algorithm. Furthermore, the output layer was added with the SoftMax activation function, and the dropout layer was set to 0.5 to avoid overfitting. Finally, the model was assembled using the categorical cross-entropy loss function and the Adam optimizer.

4. Results and discussion

The machine and deep learning models were implemented and tested on a personal Windows 10 Home 64-bit computer with an Intel(R) Core (TM) i5-7200U, Graphics-2.71 GHz, and 8 GB of RAM. The suggested methodology for this study was calculated with Anaconda Jupyter Notebook and Python 3.10; the Python libraries used are Matplotlib, sklearn, pandas, numpy, Keras, TensorFlow, etc. First, the original raw data was pre-processed to tally the frequency of each assault type, replace null and infinite values, and check for null values. Five assaults (FTP-BruteForce, DoS attacks-SlowHTTPTest, DDOS attack-HOIC, DoS attacks-GoldenEye, and DoS attacks-Slowloris) and one normal (Benign class) were present in the original data.Figure 1 illustrates how frequently each form of attack occurs. The random forest algorithm [20] was then used to pick the critical seventeen features from a total of eighty features, as shown in Figure 2. As a result, the undesirable traits were removed from processing without sacrificing vital information. After feature selection using the random forest approach, the resulting dataset had 5,43,589 rows and just seventeen most notable characteristics [20]. The crucial characteristics that were chosen using the random forest are shown in Figure 2. Models were trained and tested using the 70% and 30% of the data that were divided into train and test categories, respectively. The suggested CNN model was put into practice using the following hyperparameters: batch size=64, learning rate=0.0001, categorical cross-entropy loss, activation function='SoftMax,' and a 'Adam' optimizer for hidden nodes of 192. '30,214', '29,830', and '384' were the total trainable and non-trainable parameters, respectively.

Thus, the CNN model required "20.66" minutes for training and "0.21" seconds for testing. The obtained validation accuracy, validation loss, training accuracy, and training accuracy were, in that order, 0.9880, 0.0531, 0.9707, and 0.3591. The training and testing accuracy and loss of the implemented CNN model are displayed in Table 1. An error matrix, 10-fold cross-validation, and measures such as accuracy, precision, recall, and the F1-score, which were calculated using
Eqs. (1), (2), (3), and (4), respectively, were used to assess the CNN model's performance.

$Accuracy = TP + TN /$

$TP + TN + FP + FN$

$Precision = TP\ TP + FP$

$Recall = TP\ TP + FN$

$F1 - S\ ccore = 2 * Recall * Precision\ Recall + Precision$

Predict one and the actual is one when TP, TN, FP, and FN are all genuinely positive. Both the actual and true negatives anticipate zero. While a false negative predicts zero but is consistently one, a false positive shows one, but the essential is zero [3, 10].

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

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

.

Predict one and the actual is one when TP, TN, FP, and FN are all genuinely positive. Both the actual and true negatives anticipate zero. While a false negative predicts zero but is consistently one, a false positive shows one, but the essential is zero [3, 10].

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

[redacted]

Table 3 shows the class-specific results for the calculated CNN model's precision, recall, and F1-values on testing datasets. The training and testing sets for each epoch were subjected to the accuracy and loss of the CNN model. Because of the complexity of the situation in our case, it took forty epochs to correctly train the CNN model. data to display the accuracy and loss of the test and train. The accuracy plot has shown that the CNN model can be trained with greater than 98% accuracy using 13 number epochs. On the other hand, the testing accuracy peaked on epochs 20, 34, and 40 (Figure 4). As a result, the model has been precisely trained to identify the various

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.

assaults. As an alternative, the loss displays the overall amount of mistakes made in each training sample. We got the highest accuracy in training and testing as well as the lowest loss in this investigation (Figures 4 and 5). The accuracy of the CNN model during training and testing throughout forty epochs is shown in Figure 4. Only 2% of training was lost, which remained consistent from thirteen eras. At 20, 34, and 40 epochs, the testing loss was minimal (Figure 5). Based on forty epochs, the loss findings indicate a considerable decrease in incorrect classification. The loss curve for each of the epochs throughout the training and testing phases is shown in Figure 5. With minimal training and testing hours, the suggested CNN model has delivered accuracy for all the numerous classes.

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

The loss curve shows that there are extremely few erroneous estimations in the testing set. With forty epochs, the CNN model has therefore correctly trained. The error matrix (Figure 6) that the CNN model created, which demonstrates that the correctly categorized examples are satisfactory, has been used to assess the robustness of the CNN model. The error matrix produced by the CNN model with the selected datasets is displayed in Figure 6. Since the error matrix accurately represents true and estimated classes, it is utilized to solve classification problems. Merely 2.94% of the data in our trial were incorrectly assigned to different groups, indicating that the CNN model produced accurate results. The deployed CNN model had the greatest anticipated accuracy of 97.06% in identifying and categorizing the

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

various attack types derived from intricate datasets. Consequently, it is verified that the suggested CNN model is more accurate than previous studies [7, 8, 9, 15, 17, 18] in identifying and categorizing network attacks from intricate datasets.

5 Conclusions

In order to correctly identify and categorize network threats, the current study helps to update the raw data, select the best features from complex data using the random forest method, and construct the CNN model.The CNN model that is based on deep learning was created with the best features possible thanks to the random forest technique. The The suggested CNN model successfully recognizes and categorizes network attacks from intricate datasets. The CNN model trained and tested the datasets in a mere '20.66' minutes and '0.21' seconds, respectively. According to the experimental findings, the CNN model only needed forty epochs to achieve 98.80 and 97.07% accuracy on training and testing datasets. The data's dimensionality can be decreased, and the best features can be chosen by applying the random decreased, and the random forest method—which has the highest success rate and preserves crucial information—can be used to choose the best features. Furthermore, for complicated datasets with the highest detection rate, the deep CNN model performs better. The current study's findings can be used in real-time monitoring and reaction, enabling sectors to identify and address cyberthreats and attacks before they become factual. The results of this study are also critical to Industry 4.0 because they enable faster data-driven decision-making, real-time insights into network behavior, better responses to new problems, and improved market situation adaptation. Furthermore, this study can be very important for preserving and mitigating digital information, which is what's driving the industry's evolution.

References

[1] Rehman, S. U., and Jaber, A. N. (2020). "Intrusion detection system for cloud computing environment based on FCM–SVM." Computer Cluster Science 23: 3221–3231.

[2] Ahmadi, M., and R. Atefinia (2021). "Using multi-architectural modular deep neural network for network intrusion detection." The Journal of
3571–3593 in Supercomputing 77.

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

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

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

[6] Douzi, K., Hssina, B., Douzi, S., and Laghrissi, F. (2021). Systems for detecting intrusions that use long short-term memory (LSTM) Journal of Big Data, 8(1), 65–67.

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

[8] L. Ashiku and C. Dagli (2021). "Deep learning-based network intrusion detection system." 185: 239–247 in Procedia Computer Science.

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

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

[11] Al-Nemrat, A., Soman, K. P., Poornachandran, P., Alazab, M., Vinayakumar, R., and Venkatraman, S. (2019). "Deep learning methodology for sophisticated intrusion detection systems." 41525–41550, IEEE Access 7.

[12] Gao, Y., Wu, H. (2018), Liu, Y., Jin, Y., and Chen, J. "A new approach to semi-supervised learning for network

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

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

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

[16] In 2020, Choi, E., Shim, M., Kim, J., Kim, H., and Kim, J. "Network intrusion detection utilizing CNN for preventing denial-of-service attacks." Technology

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

[18] In August of 2021, Archana, HP, C., Khushi, Nandini, P., Sivaraman, and Honnavalli, P. "Network intrusion detection system utilizing cloud computing profound learning. In conjunction with the 2nd Forum of Women in Research, the 7th Annual International Conference on Arab Women in Computing is being held from 1 to 6.

[19] ids-2018.html on https://www.unb.ca/cic/datasets. retrieved on May 2, 2023.

[20] Zhang, Q., Wu, L., Chen, W., and Li, X. (2020). "Developing an auto-encoder intrusion detection system through the selection of random forest features." 95: 101851, Computers Security.