Convolutional Neural Network Based Model for Intrusion Detection

Olaniyi A. Ayeni, Stanley C. Ewa, Otasowie Owolafe
Cyber Security Department
Federal University of Technology
Akure, Nigeria

Abstract

In recent research works, Machine learning techniques have been effective in identifying vulnerabilities and attacks better than most traditional methods. In this Study an Intrusion Detection Model is developed using Convolutional Neural Network (CNN) for the attack features of CICIDS-2017 dataset. CNN is excellent in Computer vision, text, and object recognition. One major benefit of CNN over Machine learning algorithms is that Feature selection is done without the need for human intervention. The proposed system is based on feature extraction and learning as predicates for prediction. The various tools deployed for developing this model include Python programming language, Microsoft Excel, Jupyter notebook of Anaconda navigator etc. Evaluation of the Model's performance was done by comparing the accuracy of the Model with other Machine learning/Deep learning IDS Models, Experimental results showed that The Proposed Model's performance was higher than the performances of the Models it was compared with, in accuracy of 99.78%.

1. Introduction

In recent years, communication has gotten more digital as the internet plays a vital role, people can communicate with each other from anywhere in the world. For instance, when we use internet services or participate in live feeds, we do not know the people we engage with on the web, we stand the risk of exposure to malicious attacks and attackers. The Intrusion Detection and Intrusion Prevention system (IDPS) started with an academic paper written by Dorothy E. Denning in 1986 titled "An Intrusion-Detection Model," which led Stanford Research Instituted (SRI) to develop the Intrusion Detection Expert System (IDES). The system used statistical anomaly detection, signatures and profiles of users and host systems to detect nefarious network behaviours. IDES had a dual approach. It used a rule-based Expert System to detect known types of intrusions plus a statistical anomaly detection component based on profiles of users, host systems, and target systems. Firewalls had been very effective for countering the threat landscape of the 1990s. Firewalls process traffic quickly as they have no "deep packet inspection" meaning they have no visibility into the context of network traffic. They only can react based on protocols, ports and/or IP addresses. In the early 2000s, new threats like SQL injections and Cross Site Scripting (XSS) attacks were becoming popular and would pass right by the firewall, many organizations preferred IDS because they send an alert to the organization's administrator on detection of any nefarious traffic. Computer security revolves around confidentiality, integrity, and availability [1]. Along with other preventive security mechanisms, such as access control and authentication, intrusion detection systems (IDS) are deployed as a second line of defense [2]. The necessity of cyber physical security is rising, and traditional methods may not be effective anymore [3]. IDS schemes can be mainly classified as misuse detection schemes and anomaly detection schemes, which can be achieved by using Deep learning. Deep learning started in the 1980's, it is an unorthodox paragon in machine learning field, mainly established using Artificial Neural Networks (ANNs) and has a higher performance than the conventional Machine learning techniques. Deep learning consists of various networks such as Convolutional Neural Networks (CNNs), Deep Belief Networks (DBNs), Restricted Boltzmann Machines (RBMs), and Recurrent Neural Networks (RNNs), each of which has different capabilities and properties. These networks can carry out the learning process in unsupervised, semi-supervised, or supervised manners. Deep learning techniques have been applied in various domains such as texts, audio, wireless network processing, social network analysis and visual processing.

The Network space is continuously evolving, and attackers are developing sophisticated attack vectors to exploit network vulnerabilities. From previous research, it is obvious that most of the methods/systems proposed and that made use of KDD cup 1999 dataset have drawbacks. The work of [4] titled "Intrusion Detection Algorithm based on Convolution Neural Network" used the KDD cup 1999 dataset: This Dataset is flawed with huge amounts of redundant records, approximately 75% and 78% are duplicated in the testing and training dataset, it makes the learning algorithm biased. One

 | DOI: 10.20533/ijrfidsc.2046.3715.2023.0025

can deduce that the proposed systems used with this Dataset might present a rather lower accuracy level when used on other more recent Datasets. Another Drawback detected in the proposed system of [5] Session-Based network IDS using a deep learning-based Architecture is the Reduction of Hidden layers: This system will almost not work in a real networking environment as it requires reduction of hidden layers. Addition of more hidden layers causes backpropagation algorithm to be less effective that is improving the accuracy of predictions. In a real networking space, you'll need more hidden layers added to cover the entire network. Though much great works has been done in IDS Deep learning-based approach, but more work still needs to be done. The aim of this research is to come up with a comprehensive solution to identify / mitigate Intrusion using a recent dataset, best-choice deep learning algorithm based on the accuracy of its performance level to mitigate network attacks.

2. Literature Review and Related Works

2.1. IDS Based Detection Model

The Success rate of Intrusion Detection System is a measure of Successful Detection of attacks by the IDS, such a system should have an effective Detection mechanism to achieve a high detection rate of intrusion. Intrusion Detection System techniques or models for detecting these Intrusion can be further classified into two major Detection Models:

i. Signature-based Intrusion Detection System

ii. Anomaly-based Intrusion Detection System

These classifications were first sighted by [6] as Misuse detection and Anomaly detection.

Signature-based Intrusion Detection System: In this Model of IDS, Pattern Recognition is the focal point. Attack patterns or signatures are encoded and stored in the database as a blacklist and thereafter used to match against normal traffic to detect an attack based on the patterns or signatures in the blacklist. The blacklist should be updated with new attack patterns or signatures as they are discovered. The researchers in [7] highlight that by matching these patterns or signatures with the pattern of behavior of existing users, it is possible to identify unauthorized users or hackers. The author in [8] States that IDSs operate on the assumption that any traffic will be accepted unless it has been marked as prohibited.

Anomaly-based Intrusion Detection System: Also referred to as Statistical Anomaly-based Intrusion Detection System establishes a performance distinction between normal traffic and anomalous traffic. [9] defines Anomaly-based Intrusion Detection System as an Intrusion Detection System for detecting both network and computer intrusions and misuse by monitoring system activity and classifying it as either normal or anomalous. Classification is based on rules, rather than on patterns or signatures. Anomaly or abnormal behavior is supposed to be there but are not there and things that are not supposed to be there, are there.

Network-Based IDS: A Network-based Intrusion Detection System monitors and examines the network traffic for any suspicious activity or threats in the network [10]. It consists of a sensor (Network appliance) with a Network interface card (NIC). A sensor is used to monitor packets traveling on that network segment [8]. The network segment is used as their data source [8]. The system collects information from the network itself rather than from each separate host [7]. The NIDS audits network attacks while packets are in transit. According to [7] NIDS can be divided into Statistical anomaly IDS and Pattern matching IDS depending on how they function. An example of NIDS is Snort.

Deep Learning and Neural Networks

There is yet to be a unified definition of Deep learning but existing definitions by scholars have shared similarities about the same notion. Deep Learning is the subset of the Machine Learning which includes many hidden layers to get the characteristics of the deep network. According to [11], deep learning techniques can be classified into two models, namely deep discriminative models, and generative/unsupervised models. The deep discriminative models include Deep Neural Networks (DNNs), Recurrent Neural Networks (RNNs), and Convolutional Neural Networks (CNNs). The generative/unsupervised models include Restricted Boltzmann machine (RBMs), Deep belief networks (DBNs), Deep Boltzmann machines (DBMs), and Deep Autoencoders (DA). These techniques can be further categorized into three major classes: Deep networks for unsupervised or generative learning, Deep networks for supervised learning and Hybrid deep networks. These techniques are more efficient than the Machine Learning techniques due to their deep structure and ability to learn the important features from the dataset on its own and generate an output [12]. Deep learning is a process not only to learn the relation among two or more variables but also the knowledge that governs the relation as well as the knowledge that makes sense of the relation [13]. Deep learning is a kind of learning where the representation you form, have several levels of abstraction, rather than a direct input to output. Deep learning is a subfield of

 | DOI: 10.20533/ijrfidsc.2046.3715.2023.0025

machine learning concerned with algorithms inspired by the structure and function of the brain called artificial neural networks. Deep learning is about learning feature hierarchies with features from higher levels of the hierarchy formed by the composition of lower-level features [14].

Convolutional Neural Network

CNN is a variant of a neural network; whose objective is to learn the correct representations of the characteristics of the input data. The Convolutional Neural Network (CNN or ConvNet) is a popular discriminative deep learning architecture that learns directly from the input without the need for human feature extraction [15]. The capability of automatically discovering essential features from the input without the need for human intervention makes it more powerful than a traditional network [15]. Due to lesser parameters, CNN can be trained smoothly and does not suffer overfitting. CNN is a type of neural network that has the advantage of having a very high level of accuracy in classifying. A Convolutional neural network is a type of feedforward neural network with organized spatial operations in its layers. Each layer in the Convolutional network is a 3-dimensional grid structure, which has a height, width, and depth. Convolutional neural networks have proved to be very effective in image classification, it is also associated with other fields like object detection, text detection, object tracking, pose estimation, action recognition, image recognition, facial expression recognition, speech, and natural language processing. Many schemes have been put in place to combat Intrusion but as attack patterns evolve, it's difficult to rely on one IDS Scheme. A survey on previous studies, shows that various Deep learning techniques have shown tremendous results in Intrusion Detection system, in terms of Accuracy, precision, Detection rate and performance analysis, false alarm rate.

a. **Author(s) / Title: [16]** CNN-LSTM Neural Networks for Anomalous Database Intrusion Detection in RBAC-Administered Model.

   Objective(s): Develop and Implement CNN-LSTM Neural Networks for classifying user's role and authority.

   Methodology: Proposed CNN-LSTM Neural Networks that can classify 11 roles for intrusion detection for role-based access control-administered Relational database system with input features extracted from SQL query.

   Contribution to Knowledge: The CNN-LSTM model classifies and extracts the roles that could not be distinguished by using the conventional machine learning method.

   Limitation(s): However, the CNN-LSTM model was manually optimized. Further research is needed to automatically find the optimal parameters of the CNN-LSTM model for intrusion detection.

b. **Author(s) / Title: [17]** A Deep Learning Approach to Network Intrusion Detection.

   Objective(s): Develop and Implement a stacked non-symmetric deep auto-encoder (NDAE) for unsupervised feature learning.

   Methodology: Proposed a stacked model of NDAE and Random Forest (RF) a combination of Deep learning and shallow learning using KDD Cup '99 and NSL-KDD datasets for implementation.

   Contribution to Knowledge: It achieve high levels of accuracy, precision, and reduced training time of up to 98.81%.

   Limitation(s): The Model does not have the capability to handle zero-day attacks.

c. **Author(s) / Title: [15]**. Intrusion Detection System using Deep Neural Network and Regularization of Hyper Parameters with Adam Optimizer.

   Objective(s): Develop and Implement a Deep Neural Network model with Adam Optimizer.

   Methodology: Proposed an Intrusion Detection System Using Deep Neural Network and Adam Optimizer on NSL-KDD dataset.

   Contribution to Knowledge: The DNN and Adam algorithm model produced a better accuracy level of 96.52% when compared to the 65.2% accuracy level of the existing DNN and SGD model.

   Limitation(s): However, the accuracy in detection rate can be enhanced by using Deep learning-based models such as Restricted Boltzmann Machine (RBM), Deep Boltzmann Machine (DBM), Deep Belief Neural Network (DBNN), Spiking Neural Networks, Convolution Neural Networks etc. as classifiers and optimizing training parameters. The prediction rate of the proposed DNN and Adam Optimizer model is better for all attacks except user to root (U2R).

d. **Author(s) / Title: [18]**. Deep Stacking Network for Intrusion Detection.

 | DOI: 10.20533/ijrfidsc.2046.3715.2023.0025

Objective(s): Design and implement a Deep Stacking Network Model for detecting Network intrusion.

Methodology: Proposed a Deep Stacking Network model based on the outstanding performance of four classifiers on NSL-KDD Dataset; KNN, Decision Tree (DT), Random Forest (RF) and Deep Neural Network (DNN).

Contribution to Knowledge: In terms of training and testing time, the proposed model is acceptably higher than most algorithms, the multi-class detection accuracy of DSN reached 86.8%, the best performance.

Limitation(s): The data used in the experiment is NSL-KDD, which is an unbalanced data set. Therefore, the use of this data set for training will inevitably lead to the learning result biased towards most samples.

e. **Author(s) / Title: [19]** Deep learning approach for cyberattack detection.

Objective(s): Design and implement a framework to detect cyberattacks

Methodology: Propose a Deep Feature Embedded learning framework (DFEL) based on training deep feed-forward neural network using NSL-KDD Dataset and UNSW-NB15 Dataset to detect Internet intrusion.

Contribution to knowledge: The DEFL framework boosts classifier's accuracy to predict cyberattack and reduced the detection time significantly. DEFL when combined with traditional Machine learning algorithms KNN on NSL-KDD dataset has accuracy of 98.82% and detection time of 0.07s whereas the using KNN only had accuracy of 98.56% and detection time of 1.79s.

Limitation(s): The DEFL detection time will increase in larger dataset with many dimensions.

3. Design of the Proposed System

CNN was used as a learning model for the classification of the IDS model. The Block diagram in Figure 1 represents the system architecture when the dataset is given as an input in a supervised learning and the given input is split into Train and Test data. CNN algorithm is then applied to the trained data, we get a trained model thereafter, and validation of the trained model on the test data is performed, then finally carry out evaluation metrics for the data. CNN deep learning algorithm model is used to find accuracy of IDS from the dataset. This IDS model was implemented in python 3 with libraries of Keras, TensorFlow, Matplotlib and other necessary files.

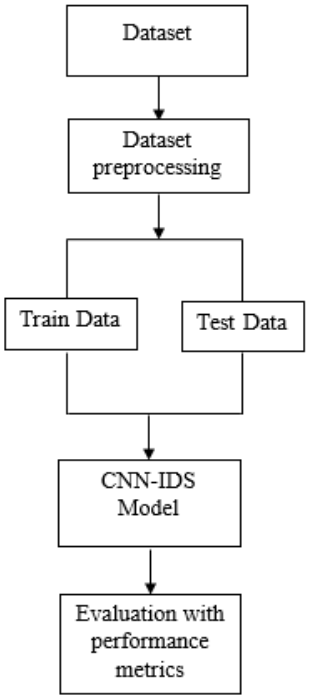

Figure 1. Proposed System Architecture for Intrusion Detection System

CICIDS-2017 dataset from ISCX Consortium is used to implement and evaluate the proposed model. This dataset is taken because it contains all the necessary features for reliable network intrusion detection. It consists of two types of traffic, normal traffic defined as benign traffic and anomaly traffic defined as attack traffic. It covers 14 types of attacks which are generally not found in other benchmark datasets. A corresponding CSV files of the entire dataset has been provided for use in Machine Learning / Deep Learning applications. The dataset contains data captured over five days starting from Monday and ending on Friday. Although it is captured for five days, each day contains a particular type of attack except for Monday that contains normal traffic flow. The attacks include DoS, DDoS, Brute Force, XSS, SQL Injection, Infiltration, Port scan and Botnet. The days and their corresponding labels are given in Table 1. The dataset contains 78 attributes or features and a corresponding Traffic label divided by each day flagged by the number of records.

The features of CICIDS-2017 Dataset not found in other datasets include:

i. Subflow Fwd Bytes and Total Length Fwd package are required to detect Infiltration and Botnet attack types.

 | DOI: 10.20533/ijrfidsc.2046.3715.2023.0025

ii. Bwd Packet Length Std attribute is required to detect DDoS, DoS Hulk, DoS GoldenEye and Heartbleed attack types.

iii. Init Win Fwd Bytes attribute is required to detect the Web Attack types, SSH-Patator, and FTP-Patator attack types.

iv. Min Bwd Package Length and Fwd Average Packet Length attributes are required to detect normal or Benign traffic.

Table 1. Analysis of CICIDS-2017 Dataset

File Name	Traffic Label	Numerical records
Monday-WorkingHours.pcap_ISCX.csv	Benign	529,918
Tuesday-WorkingHours.pcap_ISCX.csv	Benign SSH-Patator FTP-Patator	432,074 5,897 7,938
Wednesday-workingHours.pcap_ISCX.csv	Benign DoS Hulk DoS GoldenEye DoS Slowloris DoS SlowHTTPtest Heartbleed	440,031 231,073 10,293 5,796 5,499 11
Thursday-WorkingHours-Morning-WebAttacks.pcap_ISCX.csv	Benign Web Attack-Brute Force Web Attack-SQL Injection Web Attack-XSS	168,186 1,507 21 652
Thursday-WorkingHours-Afternoon-Infilteration.pcap_ISCX.csv	Benign Infiltration	288,566 36
Friday-WorkingHours-Morning.pcap_ISCX.csv	Benign Botnet	189,067 1,966
Friday-WorkingHours-Afternoon-PortScan.pcap_ISCX.csv	Benign Portscan	127,537 158,930
Friday-WorkingHours-Afternoon-DDos.pcap_ISCX.csv	Benign DDoS	97,718 128,027
Total Instances	**15**	**2,830,743**

3.1. Algorithm for Intrusion Detection System using Convolution Neural Network

The Algorithm of CNN-IDS Model described below covers the steps taking in development of the model, also it gives firsthand and detailed understanding of how the model should work.
The Following are procedures to implement Intrusion detection system using Convolutional neural network model:

i. Convert preprocessed data to image format of discrete dimensions.

ii. Import and reshape data to fit CNN model i.e., the number of samples, height of image, width of image, number of channels of the image.

iii. For greyscale input image: Initialize parameters.

for convolution layer, pooling layer, filter, and the weights of each layer.

iv. Input activation function for each layer except the fully connected layer.

v. If model fails on test data use the Regularization technique on output in fully connected layer to fit model.

vi. Use Optimization method on the output to minimize training error and time.

vii. Calculate the prediction error using Loss function to determine the performance of the CNN model.

4. Implementation and Results

The CNN-IDS Model is implemented by deploying three 2D-Convolutional layers to work on the converted 9 x 9 x 1 input image that was fed into the input layer. The Relu is used as an activation function for each convolutional layer, also padding (with zeros) is done evenly to the left/right or up/down of the input, in each convolutional layer. Flatten layer is also deployed to the feature map to transition from multidimensional array to

 | DOI: 10.20533/ijrfidsc.2046.3715.2023.0025

unidimensional array, that is from convolutional layer to the fully connected Layer. Finally, a fully connected layer is deployed as the output layer, dense layer is used in the output layer to classify the feature map based on the output of the convolutional layers. Softmax is used as an activation function in the fully connected layer for multi-class classification problems.

Classification involves the necessary steps taken in preparation of the dataset before it can be used for implementation of the CNN-IDS model. Model classification was carried out in Jupyter notebook and Libraries like Pandas, Numpy, Sklearn were used for this task.

```
_preprocessing_all(label_encoder, 2500000)
```

```
14:34:20 INFO Fill NaN in 1347 rows with average value of each class.
14:34:40 INFO Replace Inf in 2682 rows with maximum value of each class.
14:34:57 INFO Replace negative values with minimum value of each class.
14:36:57 INFO Fill NaN in 11 rows with average value of each class.
14:37:00 INFO Replace Inf in 185 rows with maximum value of each class.
14:37:02 INFO Replace negative values with minimum value of each class.
```

```
def preprocessing(df: pd.DataFrame) -> (np.ndarray, np.ndarray):
    # Shuffle the dataset
    df = df.sample(frac=1)

    # Split features and labels
    x = df.iloc[:, df.columns != 'Label']
    y = df[['Label']].to_numpy()

    # Scale the features between 0 ~ 1
    scaler = MinMaxScaler()
    x = scaler.fit_transform(x)

    return x, y
```

Figure 2. Dataset preprocessing

4.1. CNN-IDS Model Creation

The CNN-IDS model is created by implementing CNN algorithms on the preprocessed datasets. Python Libraries like TensorFlow, sklearn, matplotlib, keras are employed for this task. The CNN-IDS model is trained with the train dataset and evaluated with the test dataset. Before Implementation of the model, certain steps are taken (Figure 2) this includes reshaping the dataset before they can be fed into the neural network to avoid challenges like overfitting or underfitting, padding of the dataset to ensure the input is in the same size.

```
def create_cnn_model() -> keras.Model:
    # Creating Layers
    inputs = keras.layers.Input(shape=(9, 9, 1))
    x = keras.layers.Conv2D(120, 2, activation='relu', padding="same")(inputs)
    x = keras.layers.Conv2D(60, 3, activation='relu', padding="same")(x)
    x = keras.layers.Conv2D(30, 4, activation='relu', padding="same")(x)
    x = keras.layers.Flatten()(x)
    outputs = keras.layers.Dense(15, activation='softmax')(x)
    cnn_model = keras.Model(inputs=inputs, outputs=outputs, name='cnn')

    # Compile Layers
    cnn_model.compile(loss='sparse_categorical_crossentropy',
                      metrics=['sparse_categorical_accuracy'],
                      optimizer='adam')

    return cnn_model
```

```
# Create model
model = create_cnn_model()
logging.info(model.summary())
```

```
14:48:24 INFO None
```

```
Model: "cnn"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 9, 9, 1)]         0
_________________________________________________________________
conv2d (Conv2D)              (None, 9, 9, 120)         600
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 9, 9, 60)          64860
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 9, 9, 30)          28830
_________________________________________________________________
flatten (Flatten)            (None, 2430)              0
_________________________________________________________________
dense (Dense)                (None, 15)                36465
=================================================================
Total params: 130,755
Trainable params: 130,755
Non-trainable params: 0
_________________________________________________________________
```

Figure 3. CNN model creation

The CNN model is defined and created as shown in Figure 3; the dataset is converted to a 9 x 9 x 1 Input image represented in this order (Length x breadth x height). In the first convolution layer, filter of 120 and kernel size of 2 are convolve on the input image to get the output shape of 9 x 9 x 120. The same convolution operation takes place in the second and third convolution layers. The filter here is the number of output filters used for convolution in the convolution layer. The kernel size is an integer specifying the height and width of the 2D convolution window. These filters are selected during training of the Model. As seen in Figure 3.

4.2. Evaluation of Model Based on Performance Metrics

The CNN model is trained using the Train dataset over 30 epochs, with a batch size of 1024. An Epoch is a full cycle through a training dataset that is when the entire training dataset has passed through the network once. A batch size is the number of samples the network must work through before updating the internal model parameters. Adam optimization algorithm is used on the output feature map in training for fast computation, tunning the parameters that minimize the errors of the activation functions used to map inputs to outputs.

Model Loss is a value that repressents the summation of errors in a model. It is a measure of how well or badly a model performs. Model Accuracy is a value that indicates how well the model predicts, by comparing the model's predictions with the original values in terms of percentage. The set objective and goal is to ensure that the Model loss (shown in Figure 5) is very low that is; it tends to 0 and the Model Accuracy (shown in Figure 4) is kept at a very high level.

At the end of training the model, The Model loss and Model accuracy are 0.0045 and 0.9986 over 30 epochs.

 | DOI: 10.20533/ijrfidsc.2046.3715.2023.0025

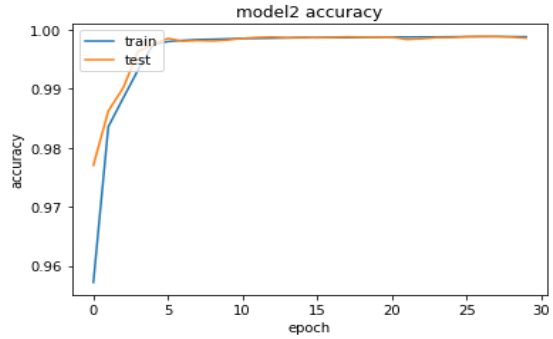

Figure 4. Model Accuracy curve

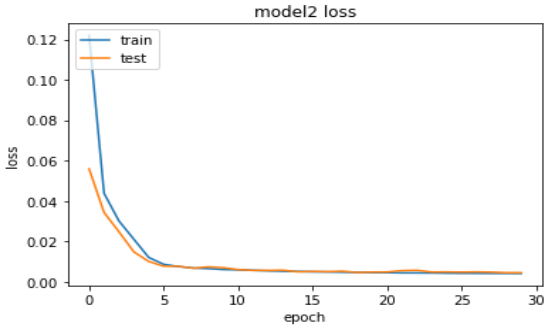

Figure 5. Model Loss curve

The results of prediction of the CNN-IDS model on the Test dataset. The Model accuracy is 99.78% and Model loss is 0.0090

Support is the total number of test samples in the dataset. The total number of samples in the test dataset is 566149

Macro avg represents the arithmetic mean between the f1-scores of the two categories, such that both scores have the same importance.

Macro avg = f1-score of preceding class + f1-score of next class / total number of classes

Macro avg is a preferred metrics if the performance of the model on all classes are considered irrespective of their sizes

Micro avg considers the number of samples per category. In other words; the greater the support the more important that category's f1-score.

Micro avg = (f1-score of preceding class x support of preceding class + f1-score of next class x support of next class) / (support of preceding class + support of next class)

Micro avg is preferred if there is a great imbalance and the performance of the minority classes are not considered. Micro avg accuracy, precision, recall and f1-score are of the same value.

From the evaluation of the CNN-IDS Model on the test dataset (Figure 6), the micro avg for accuracy is 1.00 that is 100% therefore; Accuracy, Precision, recall, f1-score of the CNN-IDS Model is 100%.

```
evaluation(model, X, Y)
del X, Y

16:30:03 INFO Evaluation:
Loss: 0.009027802385389805
Accuracy : 0.9978945255279541

16:30:22 INFO
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    454620
           1       0.72      0.39      0.51       393
           2       1.00      1.00      1.00     25606
           3       1.00      1.00      1.00      2059
           4       1.00      1.00      1.00     46215
           5       0.98      0.98      0.98      1100
           6       0.97      0.99      0.98      1159
           7       1.00      1.00      1.00      1588
           8       1.00      1.00      1.00         2
           9       1.00      0.71      0.83         7
          10       0.99      1.00      0.99     31786
          11       0.99      0.99      0.99      1179
          12       0.82      0.18      0.30       301
          13       0.14      0.50      0.22         4
          14       0.33      0.95      0.49       130

    accuracy                           1.00    566149
   macro avg       0.86      0.85      0.82    566149
weighted avg       1.00      1.00      1.00    566149
```

Figure 6. Evaluation of Model using Test Dataset

Table 2. Comparison of Different Models with the proposed CNN-IDS Model

Dataset	Author(s)	Feature Learning method	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
CIC-IDS2017	Our Work	Proposed CNN-IDS Model	99.78	86.00	85.00	82.00
KDD CUP 99	Yihan Xiao *et al.* (2019)	CNN	82.9	82.6	82.9	82.7
UNSW_NB15	Xin Xie *et al.* (2020)	CNN-GRU	84.3	83.7	84.3	84.0
NSL-KDD	Wei Wang *et al.* (2018)	CNN-LSTM	82.6	81.9	82.6	80.6
KDD CUP 99	Gunjal Somnath *et al.* (2020)	RNN	92.55	52.09	91.66	64.00

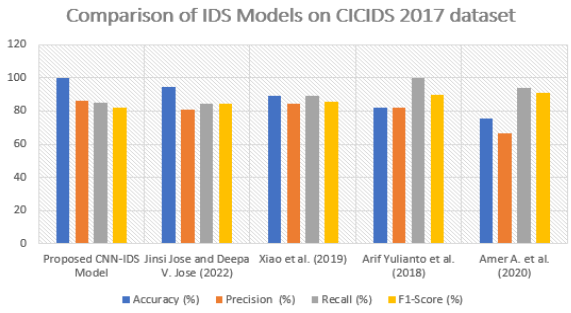

Figure 7. IDS Models That Used CIC-IDS2017 Dataset

5. Conclusion

Network security attacks have maligned the Network space for years and they are becoming Spontaneous and harder to curtail. Many hardware, software and Models created and deployed have

 | DOI: 10.20533/ijrfidsc.2046.3715.2023.0025

shown tremendous results in detection and incident response rates, but more work needs to be done, as many Malicious attacks with more advance attack vectors can easily get past most Network security mechanisms undetected and compromise security features like confidentiality, integrity, availability etc. In this study Intrusion Detection system using Deep learning, was designed and implemented based on feature extraction using Convolutional neural network on CICIDS-2017 dataset. Feature learning and modelling were achieved by analysing the Attack types in the Dataset. The proposed system was implemented in Python and performance is measured using accuracy, precision, and recall. The results obtained from the implementation of the CNN-IDS model's prediction, accuracy evaluation, when compared to other Models Table 2; Figure 7) proved that the Convolutional Neural Network model approach to Intrusion Detection performed better in accuracy 99.78%.

6. References

[1] Bishop, M. (2003). Computer security: Art and science. Boston, MA: Addison-Wesley.

[2] Ahmim, A., M. Derdour, and M. A. Ferrag (2018). An intrusion detection system based on combining probability predictions of a tree of classifiers. International Journal of Communication Systems 31(9), e3547.

[3] Barnaby Stewart, Luis Rosa, Leandros A. Maglaras, Tiago J. Cruz, Mohamed Amine Ferrag, Paulo Sim˜oes, and Helge Janicke (2017). A Novel Intrusion Detection Mechanism for SCADA systems which Automatically Adapt. doi:10.4108/eai.1-2-2017.152155.

[4] Yuchen Liu, Shengli Liu, and Xing Zhao (2017). Intrusion Detection Algorithm Based on Convolutional Neural Network. DEStech Transactions on Engineering and Technology Research. DOI:10.12783/dtetr/iceta2017/19916.

[5] Yang Yu, Jun Long, and Zhiping Cai (2017). Session-Based Network Intrusion Detection Using a Deep Learning Architecture. Lecture Notes in Computer Science DOI:10.1007/978-3-319-67422-3_13.

[6] Rod, H., Darren, M., Hai, T. (1999). An introduction to automated intrusion detection approaches. Information Management and Computer Security.Information Management and Computer Security 7(2):76-82.

[7] Jayesh Surana, Jagrati Sharma, Ishika Saraf, Nishima Puri, and Bhavna Navin (2017). A Survey on Intrusion Detection System. International Journal of Engineering Development and Research, Volume 5, Issue 2 | ISSN: 2321-9939.

[8] Joseph, S., Rod, A., Tommy, G. (2003). Intrusion detection: methods and systems. Part II. http://www.emeraldinsight.com/0968-5227.htm (Acc ess Date: 6 June 2023).

[9] Misiko N. Jacob, Muchelule Yusuf Wanjala (2017). A Review of Intrusion Detection Systems. International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online) Vol. 5, Issue 4, pp: (1-5).

[10] Tasneem A., A. Kumar, and S. Sharma, (2018). Intrusion Detection Prevention System using SNORT. International Journal of Computer Applications, vol. 181, pp. 21–24, Mar. DOI: 10.5120/ijca2018918280.

[11] Deng L. and Yu D., (2014). Deep learning: methods and applications. Foundations and Trends in Signal Processing, vol. 7, no. 3–4, pp. 197–387.

[12] Ahmed F., Jahangir U., Rahim H., Ali K., and Agha D. S., (2020). Centralized Log Management Using Elasticsearch, Logstash and Kibana. International Conference on Information Science and Communication Technology (ICISCT), pp. 1–7. DOI: 10.1109/ICISCT49550.2020.9080053.

[13] Zhang, H.; Li, J.L.; Liu, X.M.; Dong, C. (2021) Multi-dimensional feature fusion and stacking ensemble mechanism for network intrusion detection. Future Gener. Compute. Syst. 122, 130–143.

[14] Bengio Y., Boulanger-Lewandowski N., and Pascanu R., (2013). Advances in optimizing recurrent networks in 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 8624–8628.

[15] Sekhar R., and K. Thangavel (2019). Intrusion Detection System using Deep Neural Network and Regularization of Hyper Parameters with Adam Optimizer. International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249–8958, Volume-8, Issue-5S.

[16] Tae-Young Kim and Sung-Bae Cho (2019). CNN-LSTM Neural Networks for Anomalous Database Intrusion Detection in RBAC-Administered. DOI:10.1007/978-3-030-36808-1_15.

[17] Nathan Shone, Ngoc Tran Nguyen, Phai Vu Dinh, and Qi Shi (2018). A Deep Learning Approach to Network Intrusion Detection. IEEE Transactions on Emerging Topics in Computational Intelligence 2(1):41-50 DOI:10.1109/TETCI.2017.2 772792.

[18] Yifang Tang, Lize Gu, and Leiting Wang (2021). Deep Stacking Network for Intrusion Detection. Sensors 2022, 22(1), 25. DOI: 10.3390/s2 2010025.

[19] Yiyun Zhou, Meng Han, Liyuan Liu, and Jing He (2018). Deep learning approach for cyberattack detection. Conference: IEEE INFOCOM 2018 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS) DOI:10.1109/ INFCOMW.2018.8407032.

