Available online at www.qu.edu.iq/journalcm
JOURNAL OF AL-QADISIYAH FOR COMPUTER SCIENCE AND MATHEMATICS
ISSN:2521-3504(online) ISSN:2074-0204(print)

Intrusion Detection Systems Based on RNN and GRU Models using CSE-CIC-IDS2018 Dataset in AWS Cloud

Farah faiq kamel[1], Mohammed Salih Mahdi[2]

[1]Information Institute for Postgraduate Studies Iraqi Commission for Computers and Informatics

Baghdad, Iraq: ms202220728@iips.edu.iq

[2]BIT, Business Information College, University of Information Technology and Communications, Baghdad, Iraq: Mohammed.salih@uoitc.edu.iq

ARTICLE INFO

Article history:
Received: 01 /10/2024
Rrevised form: 14 /11/2024
Accepted : 02 /12/2024
Available online: 30 /12/2024

Keywords:

deep learning ;Cloud Computing;GRU (Gated Recurrent Unit); Recurrent Neural Network (RNN); CSE-CIC-IDS2018; intrusion detection system

ABSTRACT

Globally, cloud computing (CC) is becoming a necessary technological advancement. This method is a breakthrough in collaborative services and data storage. Nevertheless, the switch to CC has increased security risks, and the networks and daily interactions we engage in depend on network security. An efficient intrusion detection system is essential as attackers create new attack types and network sizes continue to rise. IDS is dependent primarily on determining whether network packets are malicious or benign. Deep learning algorithms have proved to be effective in detecting intrusions compared to other machine learning methods. In this study, we created deep learning methods to recognize attacks using recurrent neural network (RNN) architecture, namely the GRU (Gated Recurrent Unit) architecture. We use these models to handle binary and multiclass classification on the updated cybersecurity CSE-CIC-IDS2018 dataset. The recommended approach offers superior intrusion detection performance regarding Recall, accuracy, and precision. The recommended procedure yielded accuracy and precision values of 99.92 and 99.685, respectively.

MSC..

https://doi.org/10.29304/jqcsm.2024.16.41780

1. introduction

Cloud computing offers several services to users, including apps, infrastructure, and storage capacities. A cloud user may access or change hardware and software as required, mainly over the Internet. Cloud computing offers several advantages for users, although it also presents some restrictions and concerns. The obstacles of cloud computing include security, privacy, load balancing, pricing, and performance management. Among these problems, security is the most significant since user data and apps reside in the cloud. Cloud computing security encompasses rules and procedures to safeguard cloud-based data, applications, and infrastructure against unwanted access and assaults. (Lata, S et al .2022)(Aljuaid, W. A. H et al .2024) Cyberattacks pose serious security difficulties. Thus, creating a more inventive, flexible, and reliable intrusion detection system (IDS) is required. An intrusion detection system is a proactive tool that automatically detects and categorizes network-level intrusions, attacks, or security policy

∗Corresponding author :Farah faiq kamel

Email addresses: ms202220728@iips.edu.iq

Communicated by 'sub etitor'

breaches and host-level infrastructure as soon as possible (Al-Nemrat et al., 2019)(Ahmad, Z et al .2021). IDSs primarily consist of three phases.

IDS must first monitor and gather data about network flow. Second, the raw data must be cleaned by IDS and converted to the input format required for the following stage. Ultimately, to identify network traffic as normal or abnormal, a classification engine is required (Wang et al.2023) (Mahdi, H. M. S., Hassan, N. Fet al .2021). The intrusion detection dataset employed in the training model is essential to the effectiveness of deep learning applications of intrusion detection systems. Consequently, the CSE-CIC-IDS2018 dataset, which is derived from real network traffic data, can be applied using deep learning algorithms., to real-world network detection. This feature enables us to assess how well deep learning techniques perform in actual networks (Lama et al., 2023) (MS, M. 2013). This network security appliance scans all incoming and outgoing traffic for anomalous patterns that could indicate a security vulnerability in the system or network. Unlike a firewall, which is limited to searching for external intrusions, an IDS monitors the network from the inside(Farhan et al., 2020). In a recent academic study, deep learning for intrusion detection was one of the important subjects. Along with the enhancement in processing power and the quick expansion of the amount of data, Deep learning is a sophisticated subset of multilayer network-based Deep learning approaches, particularly in the large data field, and has demonstrated tremendous superiority in reduced test time and excellent accuracy We also concentrated on data processing because a lot of data can have repetitive values. Furthermore, To detect network risks, we employed a variety of models, including GRU (Gated Recurrent Unit) and RNN (Recurrent Neural Network). Last, binary and multiclass classification tasks can be used to ascertain whether traffic indicates a hostile attack.out(Lama et al.2023) (Shone et al.2018) ,(Azeez, R. A., Abdul-Hussein, M. K., Mahdi, M. S et.al.2021)

The main contribution of This research suggested a deep learning approach to create a dynamic IDS that can get around the problems of security and examine the CSE-CIC-IDS2018 on the AWS environment by using GRU (Gated Recurrent Unit)and recurrent neural network (RNN) models to reduce that limit the dimensionality of the dataset and only choose the most pertinent features to decrease false alarm reports (FAR). The paper is organized as follows. Section 2 presents the relevant literature. We provide the research methodology in Section 3 and the implementation details in Section 4. In Sections 5 and 6, we perform network traffic categorization experiments and analyze the findings. Section 7 provides the conclusion of the whole study.

NOMENCLATURE

2. Related Work

Researchers have conducted numerous studies on intrusion detection to develop better, more performant, and practical models. This section includes relevant works and the techniques currently employed in intrusion detection. Integrating big data and deep learning techniques improves the performance of intrusion detection systems. The distributed computing Apache Spark environment implements the Deep Forward-Sincering Neural Network (DNN) and two group methods, Random Forest Gradient Boosting Tree (GBT) and Random Forest (RF), to categorize the UNSW NB15 and CICIS2017 datasets. The experiment's results show high accuracy in binary classification and in classifying multiple classes using the UNSW NB15 data. The CICIDS2017 dataset yielded the results alongside the GBT classifier. DNN possessed the highest binary classification accuracy but the highest accuracy of multiclass categorization (Chockwanich et al., 2019). Shone et al. (2018) discovered a botnet attack classification, representing a well-known attack on financial transactions and banking services in a proposed system.

Using a realistic cyber defense dataset, the suggested system applied neural networks (CSE-CIC-IDS2018). Yin et al. (Zhang et al. 2023) suggest testing recurrent neural networks (RNN-IDS), a deep learning technique for intrusion detection, using the NSL-KDD dataset. They also look at the model's output for both binary and multiclass classification, considering how different neuronal counts and learning rates affect how well the suggested model works. Models using machine learning techniques like Random Forest, Naive Bayesian, J48, support vector machines, and multilayer perceptrons, among others, are trained on Weka's training set. The findings demonstrate that the model's accuracy on the KDDTest increases when the RNN-IDS contains 80 hidden nodes. In the interim, the training is at 0.5, and the learning rate is completed 80 times. The researchers in Laghrissi et al. (2021) used RNNs to construct an IDS based on deep learning. The authors of this study employed basic RNNs. Their framework was organized as follows: The training set processing section received a data set and converted categorization data into

numerical inputs. Additionally, every input was normalized by means of a scaling function. Furthermore, the data processing block feeds information to the training block for the purpose of training and model building. This investigation utilized the NSL-KDD dataset. Regarding the performance assessment, the writers considered the precision achieved using the test data as the primary standard for the best possible model. According to the results, the RNN-IDS obtained a test accuracy of 83.28% for the binary classification scheme. The training time for this model was 5516 seconds. As opposed to this, the RNN-IDS scored 81.29% (the training time for the five-way classification task was 11444 seconds). This study refrained from using any feature reduction methods that might enhance the RNN-IDS's performance while lowering training and testing times. An NIDS that uses ANNs to identify botnet attacks was presented by Kanimozhi et al. (Hiza et al. 2021). Following applying the Grid Search CV optimization method for hyperparameter optimization, MLP identified positive anomalies in the CSE-CIC-IDS2018 dataset. 99.97% accuracy and 99.91% AUC are recorded on the test set. Using the autoencoder AlexNet neural network, Dong (Basne et al. 2019) created AEAlexJNet, an intrusion detection algorithm based on deep learning. The KDD99 intrusion detection data set's experimental results show that the AE-AlexNet model has an accuracy of 94.32%. A gated recurrent unit (GRU) deep network model and multilayer recurrent neural networks were created by Xu et al. with softmax modules and perceptron (MLP) to boost the effectiveness of systems for detecting intrusions. The examinations of the KDD-99 and NSL-KDD datasets have been used to test the suggested system. The test results demonstrate that the GRU system outperforms LSTM in terms of performance for systems for detecting intrusions.

3. Research Methodology

This section explains the research findings and guides designing deep learning-based network intrusion detection systems (NIDS). The dataset CSE-CIC-IDS2018 is first preprocessed by removing all extra features, such as the date, and mapping each of the eight classes to a number between 0 and 7 into digitized values. Next, convert all of the data to [-1, 1]. Lastly, we put in place an intrusion detection system. System by the use of deep learning, as shown in Fig 1.

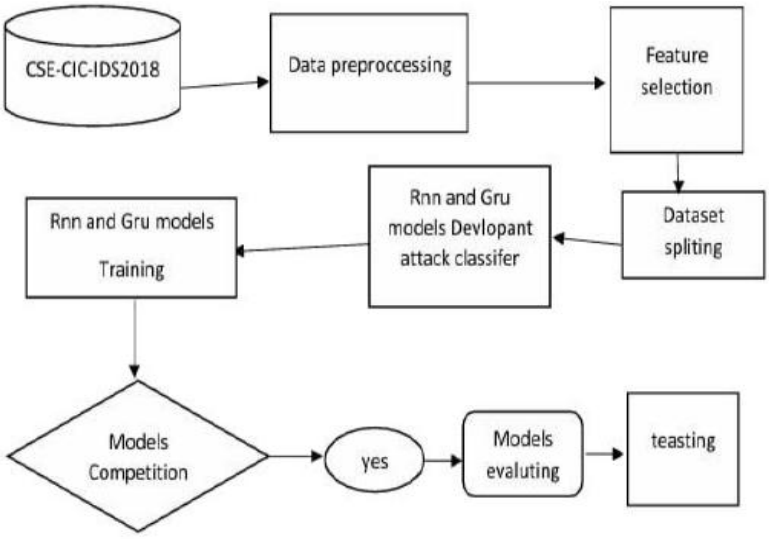


Fig.1-Workflow Diagram of Proposed Model.

3.1 (CSE-CIC-IDS2018 Dataset)

Our research makes use of the recently created CSE-CIC-IDS2018(Lata, S et .2022) real traffic data collection from the Communications Security Canadian Institute for Establishment (CSE) AWS's Cybersecurity (CIC) Infrastructure (Zhang, H et. 2023). A recent dataset of network intrusions created and released in 2018 was acquired and utilized. The datasets include malicious traffic produced by several distinct network attacks; over 5 million benign traffic samples replicating real-world behavior are included (Basnet, R. et.2019). As shown in Table 1

Table 1 - Quantity of samples and types of network traffic in every dataset

Dataset	Type of Traffic	Number of remaining samples	Quantity of Samples Removed
02-14-2018.csv	Benign FTP- Bruteforce SSH-Bruteforce	663,808 193,354 187,589	3,818 6 0
02-15-2018.csv	Benign DoS-GoldenEye DoS-Slowloris	988,050 41,508 10,99	8,027 0 0
02-16-2018.csv	Benign DosSlowHTTPTest DoS-Hulk	446,772 139,890 461,912	0 0 0
02-22-2018.csv	Benign BruteForce-Web BruteForce-XSS	1,042,603 249 79	5,610 0 0
02-23-2018.csv	Benign BruteForce-Web BruteForce-XSS SQL-Injection	1,042,301 362 151 53	5,708 0 0 0
03-01-2018.csv	Benign Infiltration	235,778 92,403	2,259 660
03-02-2018.csv	Benign BotAttack	758,334 286,191	4,050 0
Binary-class	Benign Attack	5,177,646 1,414,765	

Ten CSV files totaling 16.2 million traffic data points make up the CSE-CIC-IDS2018 dataset (Lama et al., 2023). This dataset includes seven distinct types of attacks: web, brute-force, DDoS, infiltration, botnet, and DDoS. Thirty servers and four hundred PCs comprised the compromised organizations, while fifty terminals comprised the attacking infrastructure. This dataset included the AWS network's collected traffic and CICFlowMeter-V3 machine log files with 80 extracted features(Zhang et al., 2023)(Pham, V et al .2020). Table 2 shows A subset of the traffic feature extractions.

. Table 2 - segment of the traffic feature extractions

Features	Explanation
fl_dur	Flow duration
Protocol	Transport protocol
Fl-iat-max	Maximum time interval between two streams
tot_fw_pk	total number of packets sent forward
tot_bw_pk	Total packets traveling backward
down_up_ratio	ratio of downloads to uploads
Bw-iat-avg	The mean duration between two packets transmitted across a back channel
Bw-iat-std	The mean duration between two packets that are forwarded backward

3.2 Data Preprocessing

Data preparation primarily entails modifying the source dataset to facilitate the seamless entry of network traffic data into the intrusion detection model for classification purposes. Due to input or extraction problems, a portion of the extracted dataset contains duplicate values, missing values, noisy data, infinity values, etc. Consequently, we first carry out data preparation. The six variables in the dataset—Timestamp, Flow ID, Src IP, Src Port, Dst IP, and Dst Port—were eliminated after it was discovered that they had no bearing on how attacks were classified in network traffic. When taken into account, all data features have a value of 0. During training, there would be no discrimination. It was discovered that the values of FwdByts/bAvg, FwdPkts/b Avg, FwdBlkRate Avg, BwdPSHFlags, BwdURGFlags, BwdBlk, Bwd Pkts, and BwdByts/b Avg Eight characteristics had rates with values of 0. hence, they were removed. Processing outliers was the second step in our data cleansing procedure. The data cleaning process yielded 69 columns after the removal of 11 from the original 80. The subsequent stage was eliminating rows with incorrect values.

Consequently, the rows containing values in f and -in f were removed, followed by eliminating rows with negative values in the dataset. The final step entailed converting the categorical data into numerical data and differentiating each class using One Hot Encoder on the label column (Y). Data normalization standardizes various data sizes on a uniform scale. Post-normalization, all variables have comparable scale-related impacts on the model, enhancing the learning procedure's stability and efficacy. Numerous normalization procedures exist. The most used method is min-max scaling, which rescales a feature to the defined range [0,1] by subtracting the minimum value of the feature (min) from the current value (x) and then dividing the result by the range. Defined as Equation (1):

Xi=Xi-Min/Max-Min...(1)

4. Deep Learning Models

The main objective of this study is to create a deep-learning model that can automatically identify and predict harmful and benign network flows. Two For this project, we have selected the following deep learning algorithms: Gated recurrent units (GRUs) and recurrent neural networks (RNNs) are two of deep learning's significant advancements, and their capabilities—which include multiple processing layers—allow for the learning of data's hidden representations. We apply these models to the updated cybersecurity CSE-CIC-IDS2018 dataset, handling binary and multiclass classification. The goal of this project is to increase the precision with which IDSs identify intrusion assaults in cloud environments and to improve other performance indicators.

4.1 Recurrent Neural Networks (RNNs) Model

Recurrent Neural Networks (RNNs) are a category of deep learning models characterized by internal memory, which allows them to capture sequential relationships. In contrast to conventional neural networks that regard inputs as separate elements, RNNs account for the temporal sequence of inputs, making them appropriate for tasks that include sequential data. (Shiri, F. M et al .2023) These are deep neural networks trained on time series or sequential data to create machine learning models that can make predictions or draw conclusions sequentially from sequential inputs. Recurrent neural networks (RNNs) can process information in multiple ways. In addition to looping across several levels, RNNs can momentarily retain knowledge for future use. Recurrent neural networks (RNNs) also facilitate the cycling of connections between nodes, enabling the output of one node to control the processing of input received by another. An infinite impulse response is a property of a class of networks known as "recurrent neural networks." To anticipate the output layer in an RNN, the output of one layer is fed into the input of the layer that comes before it. (Yin, C et al.2017)

4.2 Gated Recurrent Unit (GRU) Model

GRU denotes Gated Recurrent Unit, a recurrent neural network (RNN) architecture akin to LSTM (Long Short-Term Memory). Like LSTM, GRU describes sequential data by enabling information to be selectively retained or lost over time. GRU has a more straightforward design than LSTM, with fewer parameters, which may facilitate training and enhance computing efficiency. The state of the memory cell is where GRU and LSTM diverge the most. There are three gates in LSTM—the input gate, the output gate, and the forget gate—that update the memory cell state independently of the hidden state. In GRU, a "candidate activation vector" takes the role of the memory cell state and is updated via the reset and update gates. The reset gate regulates the extent to which the previous hidden state is discarded. In contrast, the update gate dictates the proportion of the candidate activation vector to be integrated into the new hidden state. GRU is a favored alternative to LSTM for modeling sequential data, especially when computing resources are constrained or a more straightforward design is preferred.

Table 3. Hyperparameter for experimentation in the proposed RNN and GRU model

Parameters	Value
Epoch	15
Batch size	265
Activation function	Leaky_relu
Loss function	categorical_crossentropy
Optimizer	Adam

The experiment's hyperparameters include the activation function, loss function, batch size, optimizer, and epoch. We carried out a thorough analysis and determined the optimal hyperparameters, as Table 3 illustrates. This RNN model uses Simple RNN layers for sequence processing, Dense layers for classification, and batch normalization to stabilize training, making it suitable for sequence classification problems. The design of this GRU model involves handling sequential data through GRU layers, producing classification output through dense layers, and enhancing training stability through batch normalization. In the first layer of RNN and GRU with 32 units, return sequences (return_ sequences=True); in the second layer of RNN and GRU with 16 units, return the last output (return_ sequences=False). After executing the layers of each model, batch normalization is applied to normalize the output. This model utilizes three fully connected layers: We employed leaky_ relu activation on 256 neurons in dense layer 1, 128 in dense layer 2, and 32 in dense layer 3. We employed the second batch normalization to normalize the dense layer's output. For multiclass classification, we employed softmax activation. The loss function represents the discrepancy between the expected and actual outputs. The Adam optimizer computes the gradients and modifies the values to minimize the loss function. This improves the functions of the RNN and GRU models. Table 3. Hyperparameters for experimentation in the proposed RNN and GRU model

5. Evaluation Metrics

We assess experimental models' performance using the ROC Curve, Accuracy, Prediction, and Recall metrics. The accuracy rate and false alert rate of intrusion detection systems are reflected in the evaluation criteria. Combining the outcomes of the model's forecast. There are four categories for the actual label: Negative falsehood (FN). It is wrong to consider a positive sample to be negative. Positive Falsehood (FP): Misreading negative samples occurs as

positive examples. Genuine negativity is known as True Negative (TN). Samples are recognized as negative samples with accuracy. Positive samples are evaluated for True Positive (TP) samples. Equations 2-4 are used to calculate these statistics.

Accuracy= TP+ TN/ TP+ TN+ FP +FN..... (2)

Precision= TP /TP+ FP.... (3)

Recall= TP +TN /TP+ TN+ FP+ FN..... (4)

6. Illustrations Results and Discussion

- ***Result for RNN MODEL of Binary Classification***

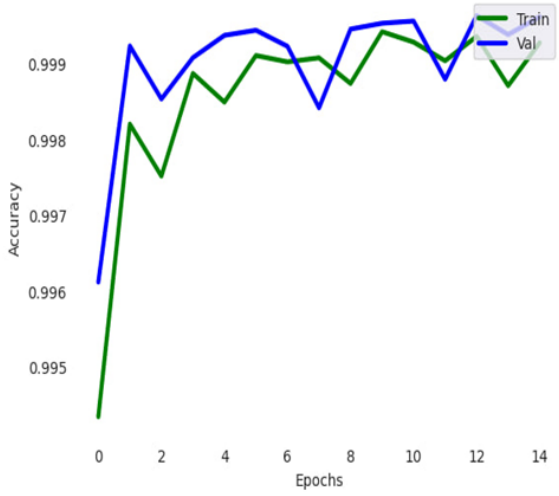


Fig. 2- Accuracy of RNN Model for binary classification

figure 2 shows that the model was trained well on the training and validation datasets with excellent accuracy. The near alignment of the two lines suggests good generalization. Green Line (Train): Demonstrates training accuracy across 15 epochs. It shows how well the model learns from the training data. Blue Line (Val): This represents the validation accuracy across 15 epochs. It assesses how well the model works with unseen validation data. Indicating that the model is not overfitting and is prepared for deployment. Both accuracies are constantly higher than 0.995, demonstrating that the model is exceptionally effective for this task.

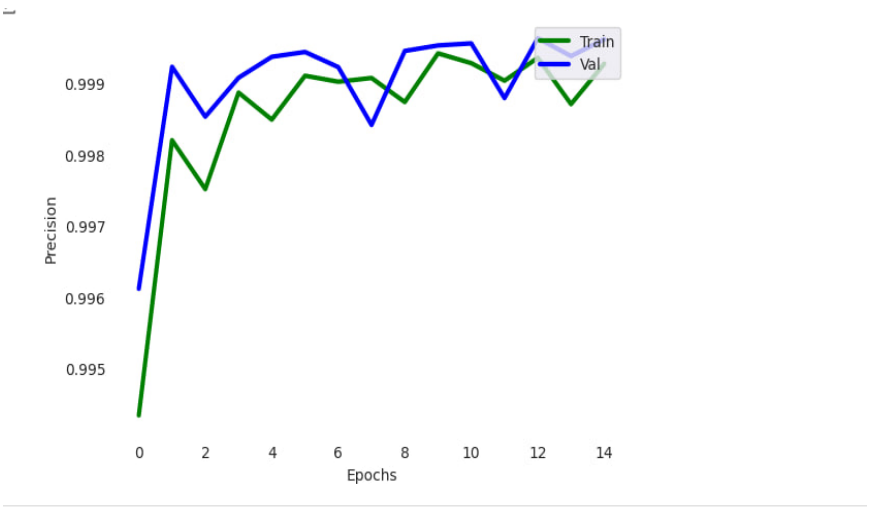


Fig.3- precision of RNN Model for binary classification.

The model obtains good precision on the training and validation datasets, as Figure 3 demonstrates. Training iterations are represented on the x-axis (epochs). The model's precision, which goes from 0 to 1, is shown by the Y-axis (Precision), and each epoch is an entire run through the training set. The proportion of genuine optimistic forecasts to the total number of actual positive and false optimistic forecasts is known as precision. The precision for training and validation stabilizes at high values, close to 1, after 15 epochs, suggesting good performance. The training and validation precisions are closely aligned, indicating that the model generalizes well and is not overfitted. Both precisions frequently exceed 0.995, showing the model successfully produces exact predictions while reducing false positives.

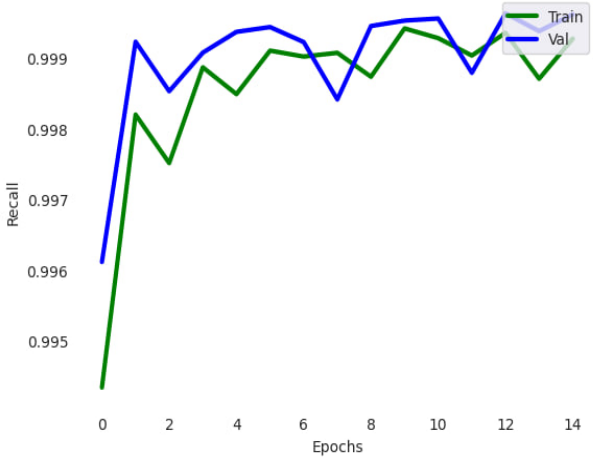


Fig. 4-Recall of RNN Model for binary classification

Figure 4 depicts the training and validation Recall across fifteen epochs for a deep learning model. Both recalls consistently exceed 0.995, showing the model is exceptionally successful at identifying positive events while avoiding false negatives.

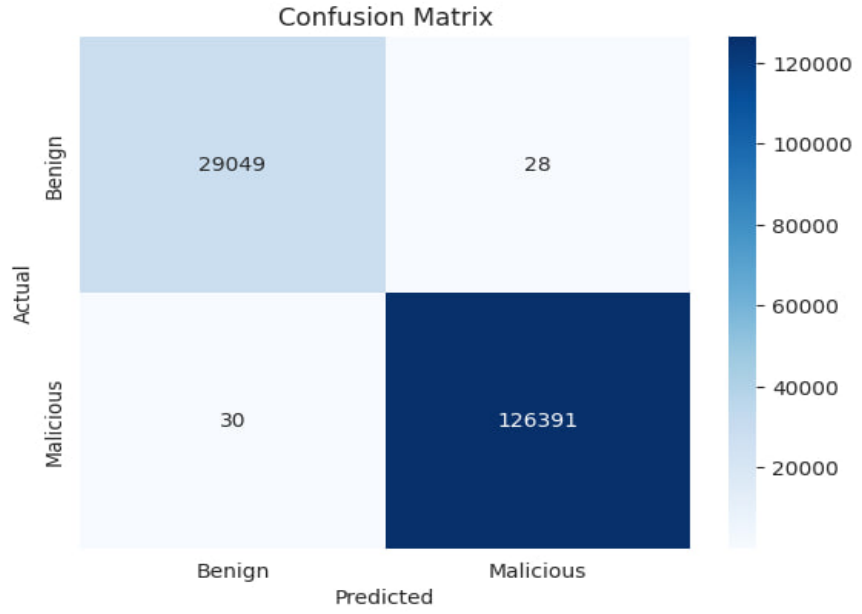


Fig.5- Final Estimator's Confusion Matrix of RNN model for binary classification

Figure 5 depicts a confusion matrix, which summarizes the prediction outcomes for a classification task. It indicates that the counts of true positive, true negative, false positive, and false negative predictions show how effectively a categorization model works. The columns show expected classes, whereas the rows show actual classes. The classifications are "Benign" and "Malicious." The values of this matrix are True Positives (TP): 126,391. The model successfully predicted "Malicious" as "Malicious," with a True Negative (TN) of 29,049. The model accurately predicted "Benign" when it was actually "Benign." and False Positives (FP) are 28. The model predicted "Malicious" instead of "Benign." A Type I mistake, often known as a False Negative (FN), is 30. The model predicted "Benign" instead of "Malicious." Also referred to as a Type II mistake.

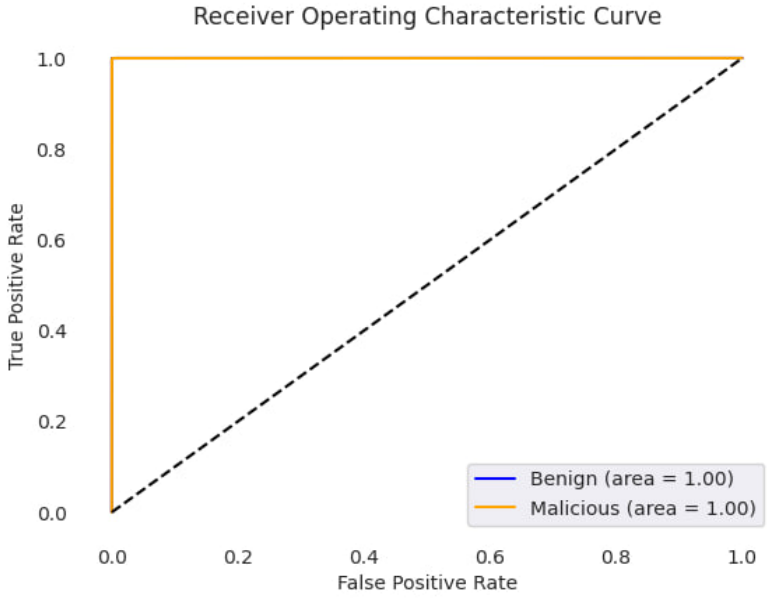


Fig. 6-Final Estimator's P-R Curve of RNN model for binary classification

Figure 6 shows that the model's ROC curve indicates that it has high discriminatory power, with flawless categorization for both benign and malevolent classifications. An AUC of 1.00 implies that the model performs exceptionally well in distinguishing between the classes without any errors. The orange line for "Malicious" and the blue line for "Benign" are both in the top-left corner, signifying flawless classification. Both classes have an AUC of 1.00, indicating that the model accurately distinguishes the classes with no overlap.

- **Result for multiclass classification.**

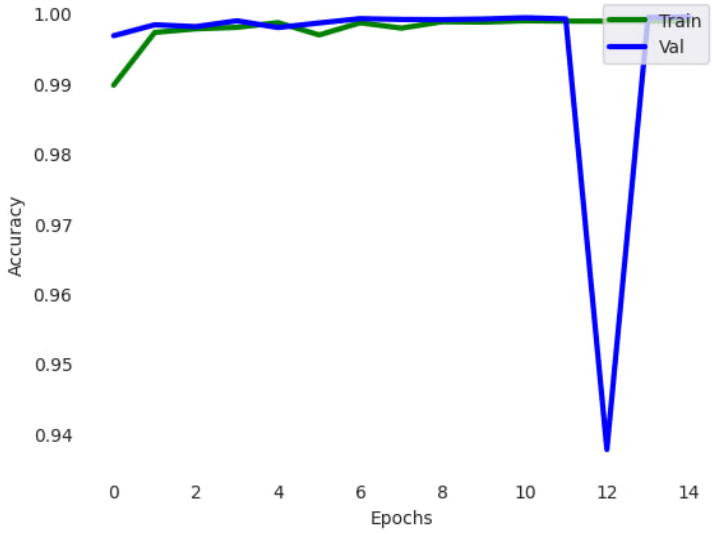


Fig.7- Accuracy of RNN Model for multiclass classification

The training and validation accuracy of a deep learning model across 15 epochs is shown in Figure 7. Training iterations are represented by the X-axis (Epochs). Every epoch is an exhaustive pass of the training set. The Y-axis shows the model's precision. Precision, which ranges from 0 to 1. Both training and validation accuracy begin high, at 1, indicating strong early performance. The training accuracy is consistent and high throughout epochs.

Fig.8- Recall and Precision of RNN Model for multiclass classification

Figures 8 and 9 depict the training and validation precision and Recall of a deep learning model across different epochs. Precision indicates how many expected positive instances are actually positive. Both training and validation precision begin high and stabilize around one, suggesting strong performance. A decline in validation precision happens around epoch 12, indicating a transitory difficulty. Precision rapidly recovered, indicating that the problem had been resolved. Recall quantifies how many true positive cases are correctly anticipated as positive. Both training and validation recall begin high and continue steady, indicating that the model is accurately collecting

positive cases. A decline in validation recall at epoch 12 mimics the precision plot, indicating the same transitory difficulty. Recall also rapidly returns, showing that the model regains its effectiveness.

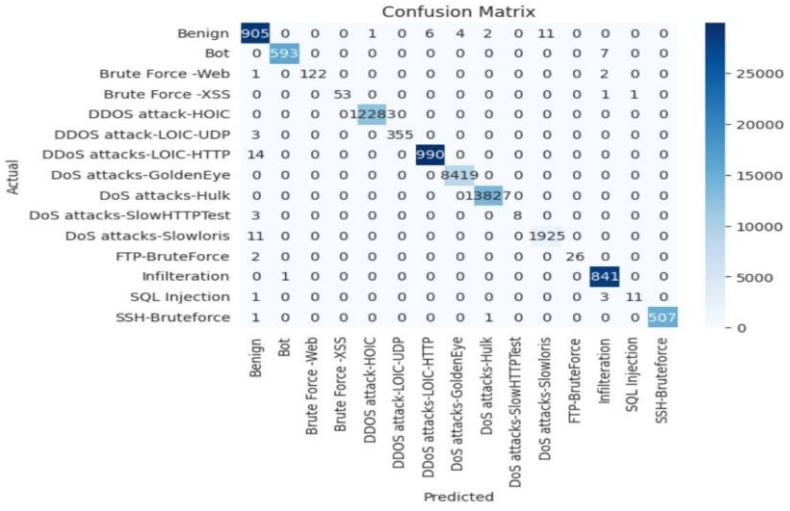


Fig.10-Final Estimator's Confusion Matrix of RNN Model for multiclass classification

Figure 10 displays the confusion matrix, which provides a thorough picture of the performance of a multiclass classification model. It compares the actual and projected labels of various network assaults and innocuous traffic. Rows (Actual) represent the valid class of the instances. Columns (Predicted) represent the predicted class of cases. High diagonal values suggest accurate forecasts. For example, "Benign" (905), "Bot" (593), "DDoS attack-HOIC" (12283), and so on. Off-diagonal values represent misclassifications. These are typically low, indicating high model performance. Some classes, such as "DDoS attack-HOIC" and "DoS attacks-Hulk," have highly correct predictions, indicating that the model can detect these classes. Misclassification is minor, with only a few "Benign" cases categorized as "DDoS attacks-LOIC-HTTP."

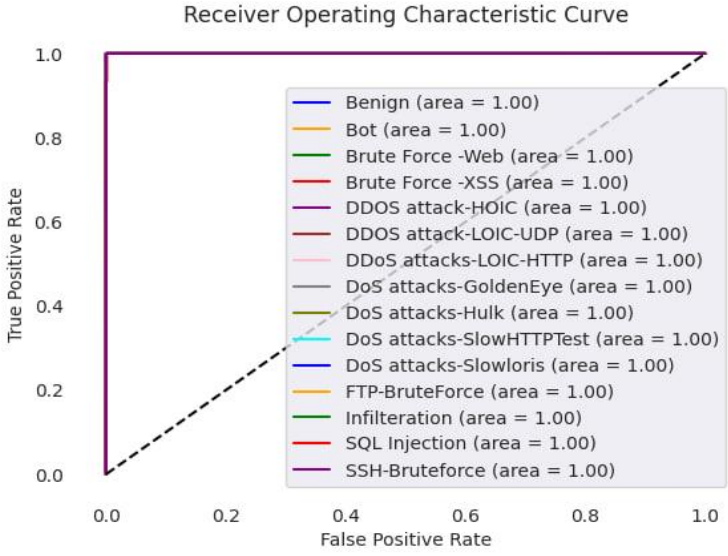


Fig.11- Final Estimator's P-R Curve of RNN model for multiclass classification

A multiclass classification model's Receiver Operating Characteristic (ROC) curve is displayed in Figure 11. The percentage of negative occurrences that were mistakenly classified as positive is shown on the X-axis (False Positive Rate). The percentage of positive events that are correctly classified is shown on the Y-axis (True Positive Rate). The trade-off between sensitivity (true positive rate) and specificity (false positive rate) is depicted by each line, which

stands for a class. The diagonal line in the illustration represents a random classifier without discrimination capacity. The ROC curve shows that the model has excellent classification performance across all classes, with an AUC of 1.00 for each. This indicates that the model successfully differentiates between various types of network traffic and attacks.

- **Result for GRU MODEL**
- **Result for Binary Classification**

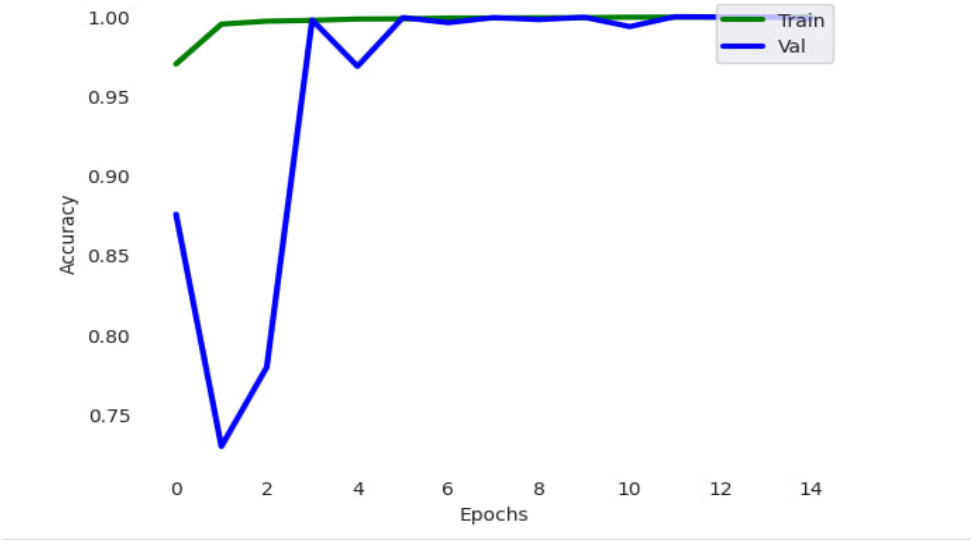
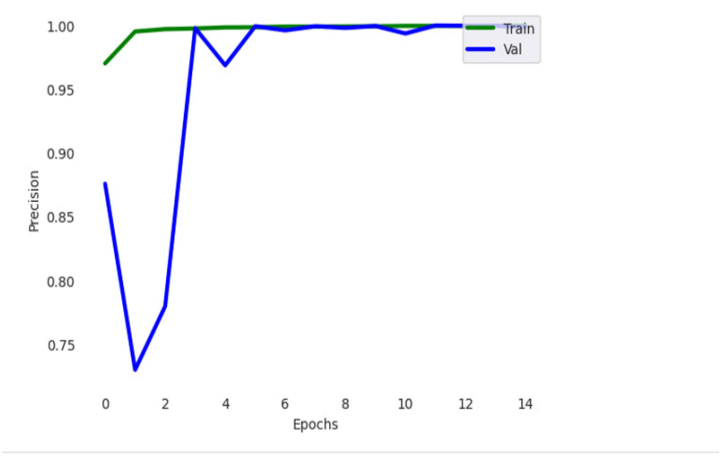


Figure 12. Accuracy of GRU Model for binary classification

The training and validation accuracy of a GRU model across 15 epochs is shown in Figure 12. The model obtains great accuracy on training and validation data after an initial period of instability. The early fall of validation accuracy indicates that the model is either overfitting or has issues with the data. That will be resolved with additional training. This picture depicts a model that soon stabilizes and reaches high accuracy following initial validation performance problems. After overcoming initial fluctuations, the model generalizes successfully.



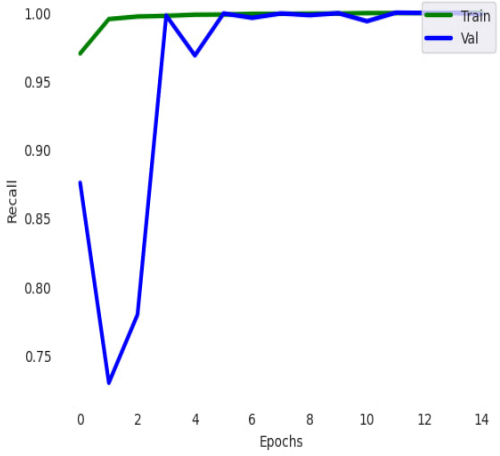


Fig.13- 14 Precision and Recall of GRU Model for binary classification.

These figures depict the GRU model's training, validation precision, and recall across various epochs. Training precision begins high and settles near 1, suggesting effective learning. An initial decrease in validation precision indicates early instability or overfitting. Validation precision improves and aligns with training precision, stabilizing at a high level. Training recall is strong at first but quickly stabilizes. Similar to precision, validation recall initially decreases, indicating early instability. Validation recollection recovers and correlates with training recall, reaching a high level. Figures 13 and 14 depict a model that quickly stabilizes and achieves excellent precision and Recall after overcoming initial validation performance issues. After overcoming initial fluctuations, the model generalizes successfully.

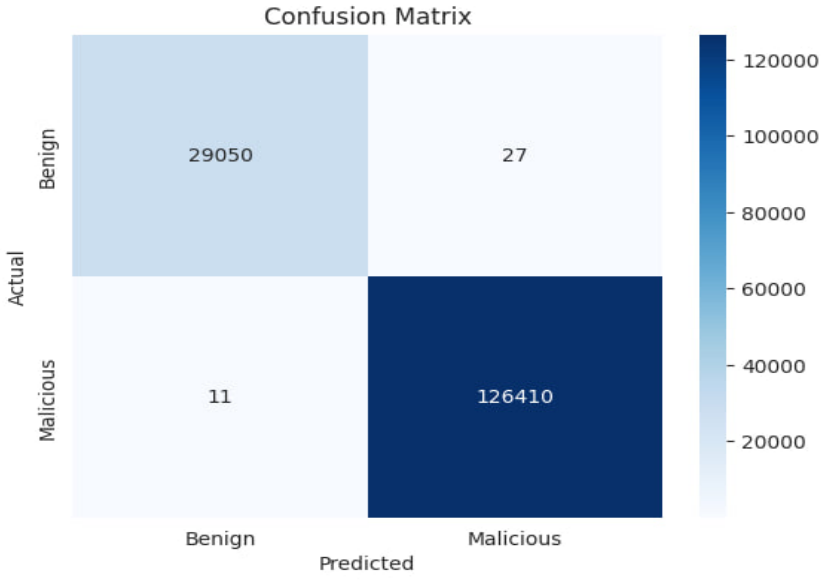


Fig.15-Final Estimator's Confusion Matrix of GRU model for binary classification

Figure 15 shows the confusion matrix of the model, which is quite effective at categorizing benign and malicious cases with very few misclassifications. This shows that the model is suitable for practically detecting network threats. The model is highly accurate, with many genuine positives and negatives. The model performs well, with low false positive and false negative counts. The model can successfully distinguish between benign and malicious cases. Values in the Matrix True Positives (TP) The model correctly predicted 126,410 "Malicious" incidents. And True Negatives (TN).29,050 The model accurately predicted "Benign" incidents. and False positives (FP)27 The model inaccurately predicted "Malicious" when it was actually "Benign." False negatives (FNs) 11 The model predicted "Benign" instead of "Malicious."

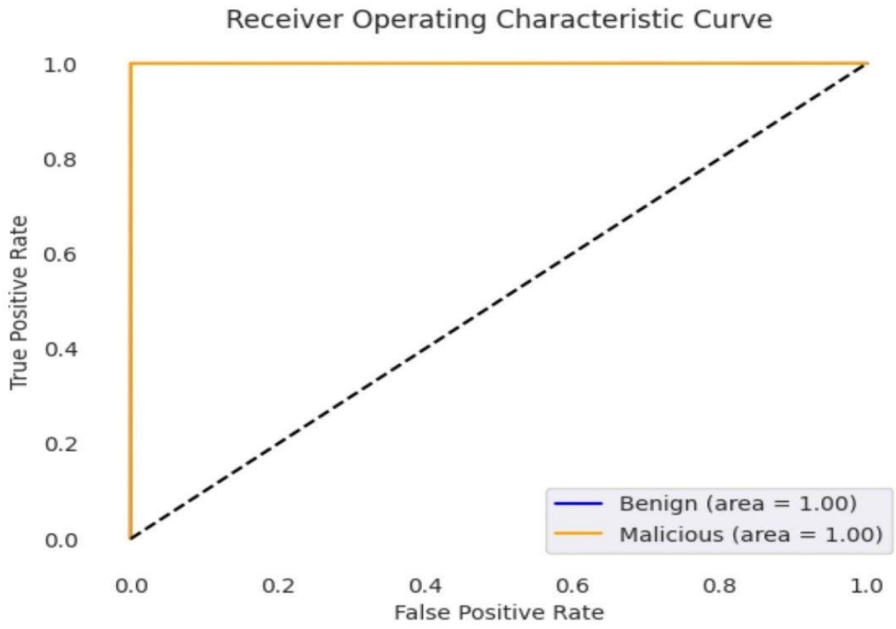


Fig.16-Final Estimator's P-R Curve of GRU model for binary classification

The Receiver Operating Characteristic (ROC) curve for a binary classification model is displayed in Figure 16. The orange line displays the model's output. It grips the top-left corner, showing exceptional classification abilities. And Diagonal Line. This is a random classifier with no discrimination capabilities. The AUC for "Benign" and "Malicious" is 1.00, suggesting flawless class discrimination. The curve demonstrates that the model is perfectly accurate, correctly categorizing all occurrences.

- **Result for multiclass classification**

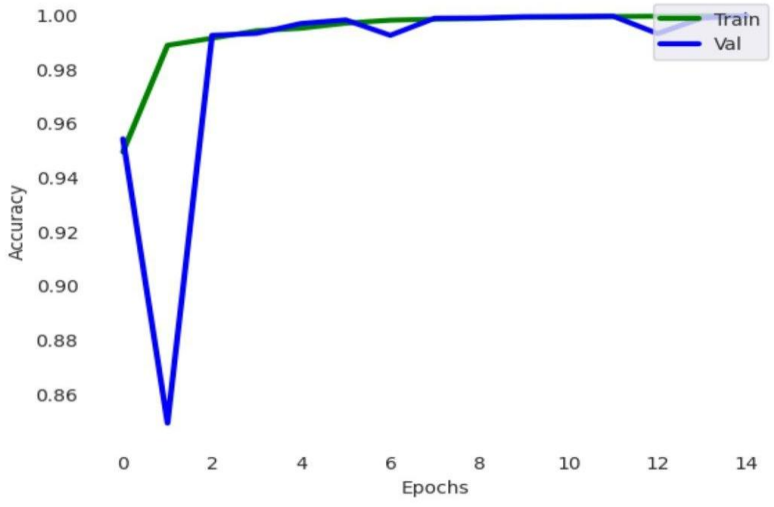


Fig.17-GRU's accuracy multiclass classification model

The training and validation accuracy of a GRU model over multiple epochs is shown in Figure 17. Training accuracy begins high and rapidly stabilizes near one, demonstrating effective learning from the training data. The initial

reduction in validation accuracy indicates early instability. Validation accuracy rapidly improves and resembles training accuracy, stabilizing at a high level after a few epochs. Despite the first swings, training and validation accuracy remain stable and high.

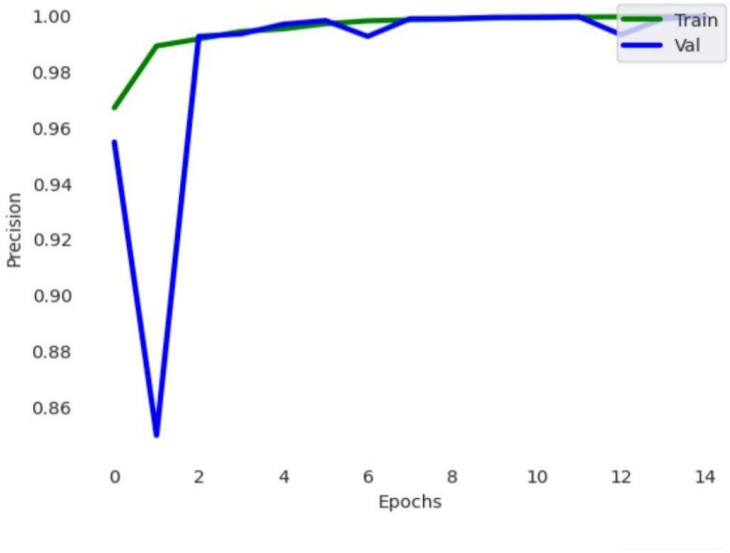


Fig.18- Precision of GRU Model for multiclass classification

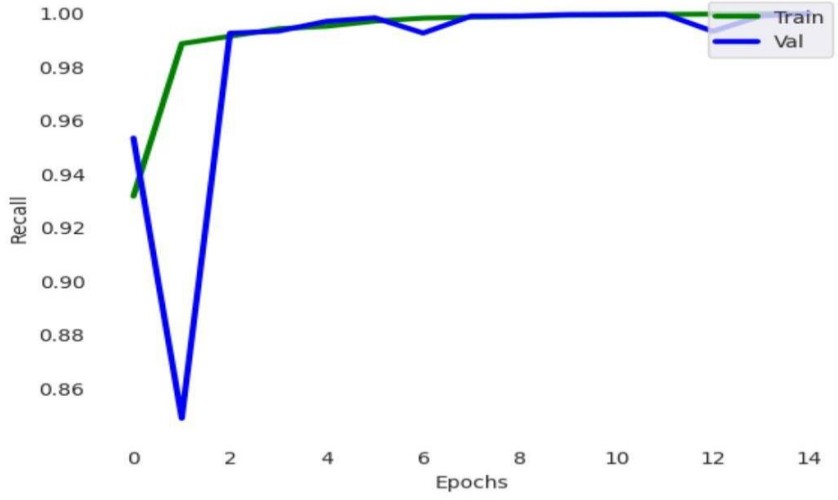


Fig.19-Recall of GRU Model for multiclass classification

Figures 18 and 19 depict a model that quickly stabilizes and reaches high precision and Recall following initial obstacles. Following an early period of instability, the model achieves excellent precision and Recall on both training and validation data. The early decline in validation precision indicates initial overfitting or data-related difficulties, which the model overcomes with further training.

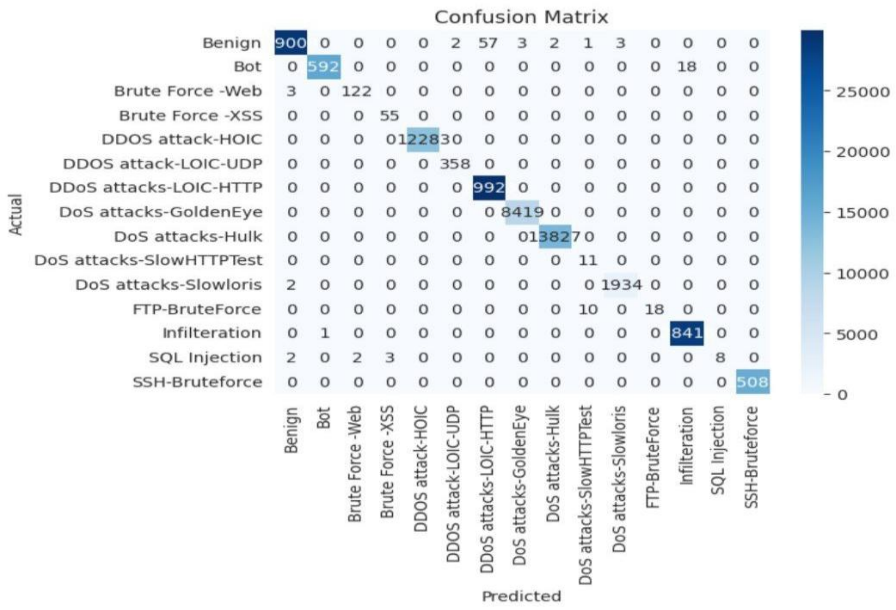


Fig.20-Final Estimator's Confusion Matrix of GRU Model for multiclass classification

The confusion matrix, which provides a detailed perspective of a multiclass classification model's performance across diverse network traffic types, is shown in Figure 20. High diagonal values imply correct predictions (e.g., "Benign" at 900, "DDoS attack-HOIC" at 12,283). Large diagonal numbers imply strong performance. Off-diagonal readings indicate misclassification. These values are often low, indicating high model accuracy. Some classes, such as "DDoS attack-HOIC" and "DoS attacks-Hulk," have very high correct predictions, demonstrating that the model recognizes these classes effectively. There are a few ambiguities. However, a few instances are misclassified (for example, some "Benign" occurrences are labeled "DDoS attacks-LOIC-HTTP").

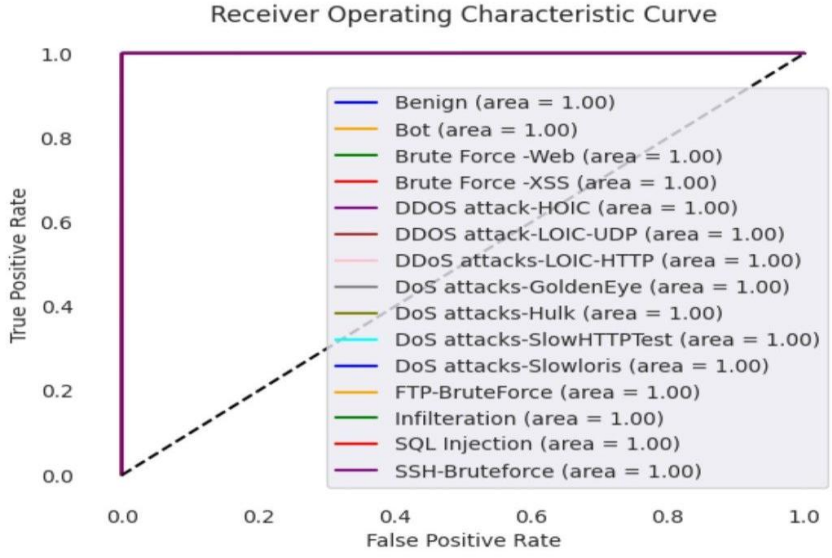


Fig.21 - Final Estimator's P-R Curve of Gr model for multiclass classification

Fig.21- The ROC curve shows the model has excellent classification performance across all classes, with an AUC of 1.00 for each. This shows that the model is extremely good at differentiating between various types of network traffic and attacks.

7. Conclusions

Cloud computing has transformed the domain of information technology via its many applications. Nonetheless, despite the use of many cybersecurity measures, cyberattacks in cloud systems are increasing. To safeguard these systems, it is essential to identify solutions and strategies to mitigate these threats. An intrusion detection system (IDS) is a vital cybersecurity defense against intrusions in a cloud computing environment. The suggested method utilizes a deep learning-based Network Intrusion Detection System model for binary and multiclass classification of the CSE-CIC-IDS2018 dataset on AWS. Classification networks, including benign traffic and fourteen distinct intrusions, are used as the training dataset. We preprocessed the data and imputed the missing values. The data's dimensionality has been decreased to simplify complexity, and the dimensionality-reduced data has been supplied as inputs to the classification module. The effectiveness of the suggested defense against cyberattacks was assessed. Recurrent Neural Networks(RNN) and Gated Recurrent Unit(GRU)classifiers were employed in our approach to classify the attack modes, producing accuracy results of up to 99%, with false positive and negative rates below 1%, for binary and multiclass classification.

Acknowledgments

We thank all the editors for their good treatment and the techniques used.

References

[1] Lata, S., & Singh, D. (2022). Intrusion detection system in cloud environment: Literature survey & future research directions. International Journal of Information Management Data Insights, 2(2), 100134.

[2] Al-Nemrat, A., Soman, K. P., Poornachandran, P., Alazab, M., Vinayakumar, R., & Venkatraman, S. (2019). An intelligent intrusion detection system using deep learning. Access Ieee, 7, 41525-41550.3. Xu, C. Z., Ye, K., and Lin, P. (2019). a deep learning-based dynamic network anomaly detection system. 3.Pages 161–176 are included in the proceedings of Cloud Computing–CLOUD 2019: 12th International Conference, Held as Part of the Services Conference Federation, SCF 2019, San Diego, CA, USA, June 25–30, 2019. International Publishing Springer...

[3] Wang, Y. C., Houng, Y. C., Chen, H. X., & Tseng, S. M. (2023). Network anomaly intrusion detection based on deep learning approach. Sensors, 23(4), 2171.

[4] Mahdi, H. M. S., Hassan, N. F., & Abdul-Majeed, G. H. (2021). An improved chacha algorithm for securing data on iot devices. SN Appl Sci 3: 429

[5.] Lama, A., and Intellectual, P. NETWORK-BASED Interruption Discovery Frameworks Utilizing AI ALGORITHMS.

[6] MS, M. (2013). Proposed block cipher algorithm with cloud computing based on keys generator (Doctoral dissertation, MS Thesis, University of Technology, Iraq).

[7] Farhan, R. I., Maolood, A. T., and Hassan, N. (2020). Execution investigation of stream put together goes after identification with respect to CSE-CIC-IDS2018 dataset utilizing profound learning. Indones. J. Electr. Eng. Comput. Sci, 20(3), 1413-1418.

[8] Shone, N., Ngoc, T. N., Phai, V. D., and Shi, Q. (2018). A profound learning way to deal with network interruption discovery. IEEE exchanges on arising themes in computational knowledge, 2(1), 41-50.

[9] Azeez, R. A., Abdul-Hussein, M. K., Mahdi, M. S., & ALRikabi, H. T. S. (2021). Design a system for an approved video copyright over cloud based on biometric iris and random walk generator using watermark technique. Periodicals of Engineering and Natural Sciences, 10(1), 178-187.

[10] Chockwanich, N., and Visoottiviseth, V. (2019, February). Interruption identification by profound learning with tensorflow. In 2019 21st worldwide gathering on cutting edge correspondence innovation (ICACT) (pp. 654-659). IEEE.

[11] Kasongo, S. M. (2023). A profound learning procedure for interruption identification framework utilizing an Intermittent Brain Organizations based system. PC Interchanges, 199, 113-125

[12] Zhang, H., Zhang, B., Huang, L., Zhang, Z., and Huang, H. (2023). A proficient two-stage network interruption identification framework in the Web of Things. Data, 14(2), 77.

[13] Laghrissi, F., Douzi, S., Douzi, K., and Hssina, B. (2021). Interruption discovery frameworks utilizing long transient memory (LSTM). Diary of Large Information, 8(1), 65.

[14] Hizal, S., ÇAVUŞOĞLU, Ü., and AKGÜN, D. (2021, June). Another profound learning-based interruption identification framework for cloud security. In 2021 third Worldwide Congress on Human-PC Connection, Streamlining and Automated Applications (HORA) (pp. 1-4). IEEE.

[15] Basnet, R. B., Shash, R., Johnson, C., Walgren, L., and Doleck, T. (2019). Towards Distinguishing and Grouping Organization Interruption Traffic Utilizing Profound Learning Systems. J. Web Serv. Inf. Secur., 9(4), 1-17.

[16] Yin, C., Zhu, Y., Fei, J., & He, X. (2017). A deep learning approach for intrusion detection using recurrent neural networks. Ieee Access, 5, 21954-21961.

[17] Shiri, F. M., Perumal, T., Mustapha, N., & Mohamed, R. (2023). A comprehensive overview and comparative analysis on deep learning models: CNN, RNN, LSTM, GRU. arXiv preprint arXiv:2305.17473.

[18] Aljuaid, W. A. H., & Alshamrani, S. S. (2024). A deep learning approach for intrusion detection systems in cloud computing environments. Applied Sciences, 14(13), 5381.

[19] Ahmad, Z., Shahid Khan, A., Wai Shiang, C., Abdullah, J., & Ahmad, F. (2021). Network intrusion detection system: A systematic study of machine learning and deep learning approaches. Transactions on Emerging Telecommunications Technologies, 32(1), e4150.

[20] Pham, V., Seo, E., & Chung, T. M. (2020). Lightweight Convolutional Neural Network Based Intrusion Detection System. J. Commun., 15(11), 808-817.