Cyber Security Intruder Detection using Deep Learning Approach

Cyber Security Intruder Detection using Deep Learning Approach

Tariqul Islam[1], Md. Mosfikur Rahman[2], Mohd. Saifuzzamam[3], Md. Ismail Jabiullah[4]

[1, 2 ,4] Daffodil International University, Dhaka, Bangladesh.
[3] Bangladesh Japan Information Technology Limited, Bangladesh.

[1]tariqul15-2250@diu.edu.bd, [2]mdmosfikurrahman.cse@gmail.com, [3]mohdsaifuzzaman.cse@gmail.com,
[4]drismail.cse@diu.edu.bd

Abstract. Intrusion detection systems (IDS) are among the most promising approaches for securing data and networks; through the years, numerous categorization algorithms have been utilized in IDS. In recent years, as the alarming increase in computer connectivity and the substantial number of applications associated with computer technology have increased, the challenge of cyber security is constantly rising. A proper system of protection for numerous cyber-attacks is also required. This is how incoherence and attacks in a computer network are detected and IDS developed, which could play a possible role in cyber security. The authors used the CICIDS2017 dataset to meet this objective. It is the 2017 set of the Canadian Cyber Security Institute. The authors propose an IDS based on the deep learning technique to increase safety. The purpose was to use a neural network classifier to predict the network and web attacks.

Keywords: Cyber Security, Artificial intelligence, Deep Learning, Cyberattack, Intrusion detection system

1 Introduction

The number of applications that stream services to their consumers has expanded dramatically in recent years. Because the apps run on the service provider's cloud servers rather than the local terminal, this service requires the user terminal's minimum installation and computational resources [1]. Many organizations have begun to establish their streaming services, seeing the clear benefit of delivering better service to clients who have no high-end equipment access. “On the other side, the large network-level interchange of data cloud servers to local users raises the assault area. Anonymous may apply several tactics, such as DDoS, port scan, and penetration to steal crucial data and render services unavailable to users” [2]. A significant issue that has to be addressed to avoid these invasions is the creation of reliable and effective IDS for cyber security. Systems of intrusion detection are ancient ideas; P. Anderson characterized an IDS as a program that explores informs management for suspicious activity or system regulatory breaches [3]. Intrusion detection systems most traditionally rely on the attack signature database of experts to operate in conjunction with the predetermined software judgment criteria for intrusion detection [4]. “The author argued that it’s simple to design and Understand IDS-based signature if unusual target behavior is known to be network performance. However, cyber-attacks have gotten increasingly corrupted in recent years, particularly assaults on systems that store or manipulate sensitive information” [5]. Without frequent updates, the expertly built attack database will rapidly become obsolete. Another key problem with intrusion detection systems based on signatures is that these systems are not wide enough to recognize new attacks with signatures in the signatory database. Briefly, large overhead storage for the attack signature database comprises signatures of all known attacks that make it hard to build or distribute. Furthermore, It may be computationally costly to compare incoming data flow with fingerprints in the data collection. “The Architecture of ANN is typical to predict and classify. ANN has various characteristics which make it particularly effective for network intrusion detection” [6]. For starters, ANN excels with a great variety of input properties in non-linear modeling data, for instance, network packets. Secondly, the future propagation of the ANN or forecasts is quick once trained. If the IDS model is aligned with network traffic, this is crucial to network performance. ANN is trained to provide an overall solution to one job using a vast dataset. Traditional intrusion-detection systems based on signatures, on the other hand, identify intrusions using manually set and comprehensible criteria. The ANN technology is strongly mathematical by using a stochastic gradient descent strategy to convey the mistake [7], [8]. ANN's training phase, no predetermined rules are necessary. This means that developers do not need to be cyber security professionals to train ANN-based intrusion detection systems. Moreover, as ANN-based IDS decision-making algorithms are generalized by all known attacks, upcoming attacks with comparable characteristics can be recognized about existing attacks. On the other hand, they will miss creative assaults due to a lack of information about their distinctive signatures. “This article presents an IDS based on the Convolutional Neural Network architecture. Unlike other previously recommended CNN IDS which focus on a class or a subset of classes” [9–11], it is good to identify unique and well-known methods of attack on the dataset in multiclass classification. Moreover, compared with the advanced, multiclass-based CNN IDS like that of Potluri for classifying the dataset of UNSW-NB15 [12].

2 Review Works

In [13] categorized several IDS models according to approaches for detection. The IDS uses statistically driven ways to construct a distribution model for innocuous traffic and recognizes low likelihood events of possible attacks. On the other side, a knowledge base is developed which reflects the large traffic profile used by knowledge-based approaches. Each activity that differs from the conventional form is therefore designated as an invasion. Finally, machine-learning algorithms are used for intrusion detection systems. These employ huge quantities of data to model some components of any kind of attack and then to categorize traffic according to the knowledgeable characteristics. “A data collection survey is also available for intrusion detection systems. Some of the public data sets were explored including Knowledge Discovery Databases (KDD) Cup'99, CAIDA, Network Security Laboratory-KDD (NSLKDD), and CICIDS2017, and feature selection and kind of computer assaults were also reported in the comparative analysis of these IDS datasets. Finally, the authors provided classification findings for the selected datasets, based on their past study. Their model which mixes a neural network with a payload classifier with a Multilayer Perceptron (MLP) obtained an exactness of 95% at CICIDS2017” [14]. “Several database development techniques were offered to boost the performance of the model in the detection of current network intrusions to attain a useable multiclass classification accuracy. The CICIDS 2017, semi-controlled K-means method was developed by Yonghao Gu et al. for the detection of DDoS attacks” [15]. In addition, they have established a hybrid selection strategy, which did not use unlogical characteristics as input to the model, to prevent a 'dimensionality curse. The features accessible are added to your functional selection algorithm. Before the algorithm outputs, chosen characteristics are handled, including data normalization, ranking, and feature finding. Finally, the feature selection technique achieved a rate of detection is 96.50 percent and 30.5 percent false positive rate. Deep learning practices like neuronal networks, due to their potential to generalize more sophisticated task data patterns, have become more successful solutions for categorization problems in recent years. The authors of [16] “examined anomaly analyzes of the intrusion detection utilizing K-Nearest Neighbors and Deep Neural Network (KNN) and the Deep Neural Network (DNN). CICIDS2017 was utilized as the database for model performance simulations in the study. They concluded that DNN was significantly higher than close neighbors. Their DNN, for the instance, is 96,427 percent accurate, which is considerably more than 90,913 percent for the closest neighbors. In addition, the overall calculation time of the two models was investigated." "The 110(s) CPU time of the DNN is below 130(s), which shows that it has a shorter overhead time than nearest k neighbors. DNN's overhead time is below that of 130(s). The study on deep learning models for cyber security within IoT (IoT) networks is being continued by Monika Roopak, Gui Yun Tian, and Jonathon Chambers" [9]. "A hybrid LSTM and CNN model employing DDoS samples from the 2017 CICIDS evaluates the performance of MLP, Long Short term Memory (LSTM), CNN. The model LSTM reached 98.44% accuracy, followed by the 98.14% precision CNN and the 97.41% precise hybrid model. The MLP model finally reached a precision of 88.47% in its simulation. The authors also compared the results to various approaches to machine learning. After a simulation, all the deep learning models assessed, except for MLP, have outperformed machine learning models like SVM, Bavaria, and Random Forest. To evaluate the performance of Naive Bayes, SVM, and C NN-based classifier have used the CICIDS2017 dataset" [10]. The study focused on the binary classification performance of the model in the dataset for each attack class. The raw data set for the CICIDS2017 was utilized to train the models which include various network activity sub-datasets all day long. Each sub-data set comprised mainly of just 13 attack categories and was trained and tested by the authors. The CNN-based IDS proposed in [11] The authors are using deep learning approaches to construct a 2018 CICIDS dataset CNN model that contained but had a higher sample size with the same characteristics as 2017 CICIDS. The study models have been trained and evaluated using CICIDS2018 sub-datasets covering a sub-set of network traffic categories. Consequently, the models were simulated instead of all at once for a multiclass classification for some classes in the data set. Another deep learning model, which is common when time series of data are employed as input, is that of CNN-basic IDS that can be bigger than the recurring neural network. "The proposed CDN model has obtained 96,77% accuracy of the CICIDS 2018 sub-dataset in the benign and DoS samples. In this research, however, the RNN model tested attained an accuracy of 82.84 percent in the same dataset, which was far lower than the CNN model. A new hierarchical IDS based on Decision Tree and rules-based models was introduced in this study" [17]. CICIDS2017 was also used as a data set to examine the performance of the model. "The proposed model combines the first stage Reduced Error Pruning Tree (REP Tree) and JRip. To classify traffic as malicious or benign, the input properties of the data set are used as input. To get the final classification result, a Forest PA classifier then uses the output of the two classifiers in the first stage, coupled with input features of the initial data set" [18]. Their concept has been successful in virtually every traffic class at CICIDS2017. They also tested the performance of their recommended model with 11 renowned classifications to demonstrate its classification capability. They had the lowest false alarm rate on benign road traffic, overtaking the other 12 classification models in 7 assault categories. Due to its good overall classification performance, this model is competitive in CICIDS2017. Therefore, for the evaluation of the proposed model performance, in the findings component of this survey, the recommended IDS model was compared to its unique hierarchical IDS. Traffic records may be retrieved rapidly and network problems can be detected utilizing technology. Traditional network analysis, IDS and malicious activities have limited recognition and reaction capability in dynamic and long-term series [19]. Canadian researchers offer an IDS for the detection of intrusion in the network based on the Convolutions Neural Network. Nine other known classifiers were compared to the proposed model [20]. Botnets are one of the most severe cyber security concerns every day for organizations. This article constitutes a detailed evaluation that thoroughly explores the botnet problem. It describes all possible techniques of detecting botnet [21].

3 Research Methodology

The authors are using the most recent Intrusion Detection Evaluation Dataset (CICIDS2017) [22]. This is more of a proof of the concept for the usage of FFBP neural network classifiers in IDSs than a final working product. The dataset contains network traffic data during regular traffic and execution of different attacks. The authors need standard Python ML tools such as modeling goal Pandas, Scikit learns, TensorFlow 2.0, and Keras. Pipeline Main part of the ML pipeline is the neural network classifier built with TensorFlow 2.0. Data is contained in 8 different CSV files, each having additional attack data at other times. So the first thing the authors must do is merge all the data from files into one Pandas DataFrame. They are reading all the CSV files into data, frames and putting those DFs into one list. Next, the authors show the number of rows and columns for each table. The authors already established that all tables have the same number of columns. That is why the authors loop over all given tables and compare them to all others combining all tables into one dataset. This is possible since all tables have the same columns, as the authors checked. By checking the shape of the dataset, the authors can confirm that concatenation has been successful.

3.1 Data Analysis

Some general info about the dataset. It contains roughly 2.5 million records across 79 columns. Data consists of mostly int64 and float64 types, except three attributes of the 'object' type. First of all, the authors determined the kinds of attacks they want to use in the dataset. These attack types are DoS DDoS and Port Scan. The authors created a CSV file from the dataset covering these three attack traffic and BENIGN traffic. Reading the data from the dataset file, combining DoS attacks, reducing the noise, finding NaN values, and assigning the column average, making the data type int64 and float64. The dataset contains network traffic data during different attacks, represented with values like port numbers, IP addresses, packet lengths, SYN/ACK/FIN flag counts, packet size, and others. Further examination reveals that the dataset contains 15 labels, "including BENIGN, DDoS, PortScan, Bot, Infiltration, Web Attack Brute Force, Web Attack XSS, Web Attack SQL Injection, DoS Hulk, DoS GoldenEye, Heartbleed, FTPPatator, SSHPatator, DoS Slowloris, DoS Slowhttptest, Labels represent network/web attacks and BENIGN" [32] state which is the network traffic during a typical business day. Most records in the dataset are of DDos and DOS Hulk attacks. This might pose a problem later in model training, considering minimal data for most attacks. This information will significantly influence model selection.

3.2 Preprocessing & Data Cleaning

At this stage, from Figure 1, the authors just clean the code containing data. The authors go through renaming columns, removing NaN and non-finite values (-inf, inf) to get the data ready for visualization and model training. Removing whitespaces in column names. Then the authors can see that the 'Label' column contains some weird characters. The following snippet uses regular expressions to replace quirky characters with Dundas. Replacing 'Label' column values with new readable values. Checking if there are any NULL values in the dataset and which column/s contain NULL values. And also how many NULL values this column contains. Next, remove all NULL values, removing rows that have NULL values. Additional attacks Check if the number of rows that have been eliminated is equal to the Null value number. Considering that only 334 rows contain NULL values in the entire dataset, which makes about 0.01%, the authors can safely remove all NULL rows without spoiling the data. Subsequent checking if all values are finite. Checking what column/s contain non-finite values and how many non-finite values each column contains. Same as before, since there is a small number of non-finite values, the authors can safely remove them from the dataset without spoiling the dataset. Replacing infinite values with NaN values. The authors can see that now the authors have Nan values again. Bringing the Labels back into the dataset before deleting Nan rows and removing new NaN values. Converting the final version of the data to CSV format, saving a cleaned dataset.

3.3 Data Visualization

So, by now, the authors know the author's dataset has 78 features and is split into 15 categories (14 attacks and 1 "normal" state). The next step is to try and visualize what the dataset looks like in feature space. To do this, the authors will employ the principal component analysis (PCA) to decrease dimensionality before passing the reduced dataset to t-SNE (t - Distributed Stochastic Neighbor Entities) for visual representation in 2D space.

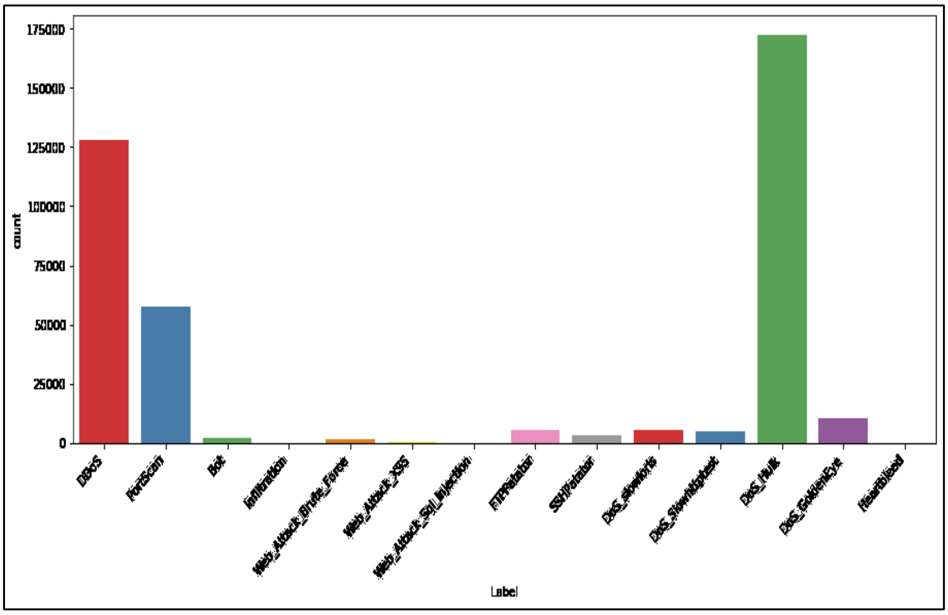

Fig. 1. Importance Feature

The authors are going to pick 10,000 random rows from the dataset for visualization purposes. Setting the random seed for reproducibility of results. Performing the principal component analysis. With just 19 components, the variance ratio remains 99%, which is excellent. Then just computing t-SNE. From Figure 2, the authors can see the distribution of data in 2D space. Attacks are not spatially well separated from a normal state. Clusters of seizures can hardly be seen. Instead, they are found in the same place as the "normal state" data points. This insight leads us to conclude that the ML model will probably have some issues with this kind of data. ML models will have to be chosen with this in mind.

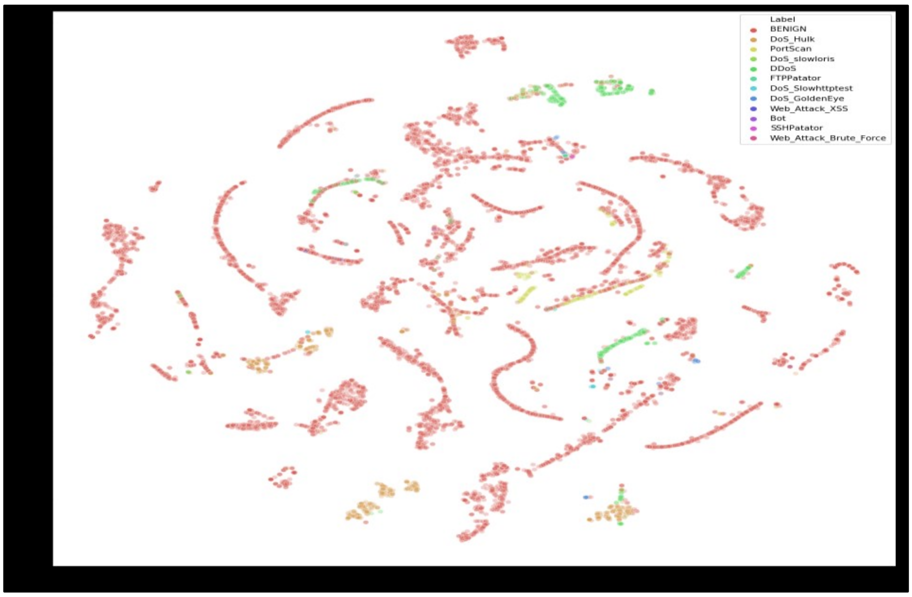

Fig. 2. Feature Implementation Visualization

3.4 Data Preparation

Final data, preparation steps are taken before the authors use the data for model training and testing. The following few steps process for scaling the data into the size adequate for the ML algorithm. Splitting the dataset into features and labels. For mounting and scaling the data, the author's RobustScaler class uses form Scikit Learn. RobustScaler is used to preserve outliers in the data. Checking if scaling has been successful. Label encoding is done when the dataset contains categorical values (ex. 0-5, A/B/C, 55+). Turn absolute values into numerical values by replacing data categories with integers starting with 0. No need to do previous operations. To convert this into numerical values, the authors will use the 'LabelEncoder' class from Scikit Learn. Labels have been replaced with integers. Splitting the data into training and testing sets is the final stage in data preparation. For this, a Scikit Learn function does all the splitting for us. This step is essential so the authors can have representative data for evaluating the author's model. Both train and test samples should contain similar data variance. The next step is to split training and testing data. For this, the authors will use Scikit Learn's training test splitting function.

4 Proposed System

For completing all steps, the authors chose to use a neural network. Specifically, the multilayer perceptron, more specifically, feedforward neural network multiclass classifier with the "backpropagation algorithm". NN will be used to classify 14 different attacks and one normal state. The author's TensorFlow Sequential model consists of three layers. There are three layers: one visible, one concealed, and one output. The input layer has 78 neurons, one for each feature. The hidden layer has 67 neurons, and this number is calculated by the formula [1] 2/3 the number of input neurons + several output neurons. The output layer has 15 neurons, one for each class the authors predict. For activation functions, the authors used standard procedures for multiclass classification tasks - ReLu for the hidden layer and "softmax function" for the output layer. Figure 3 shows the entire proposed system.

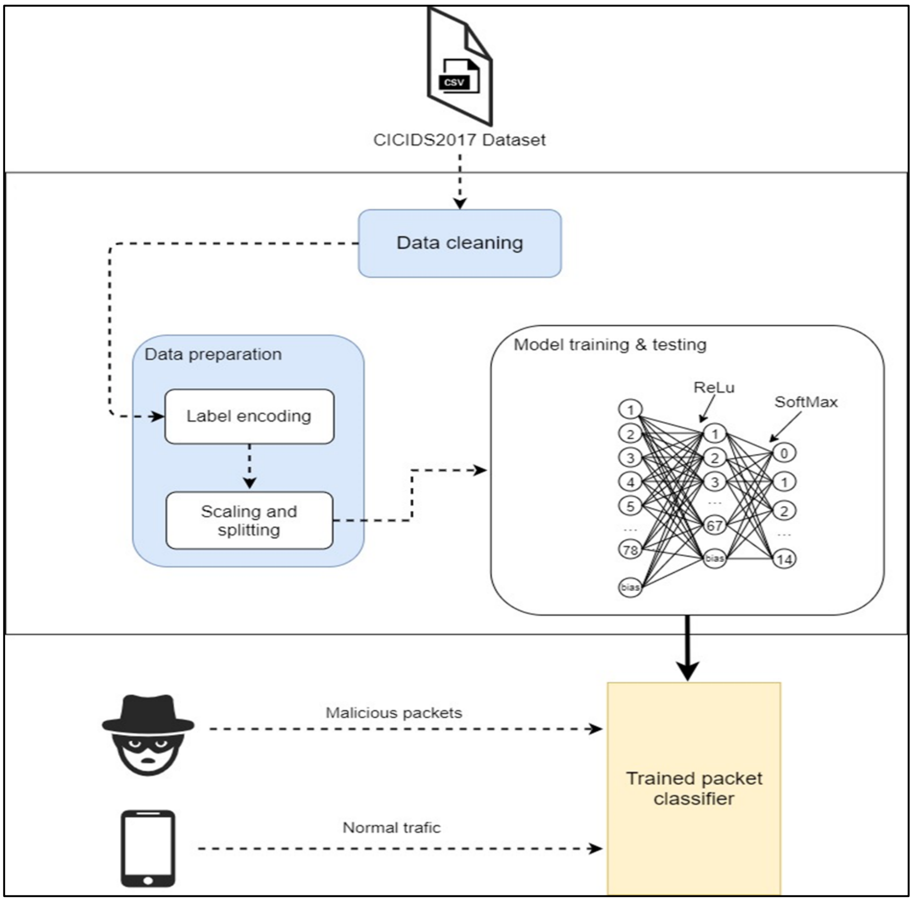

Fig. 3. Proposed System

The authors use the Dropout parameter set to 0.2 for randomly shutting off 20% of neurons in each learning iteration. This technique is used for decreasing overfitting, thereby increasing network accuracy. For learning rate optimization, the authors used Adam optimizer. The loss function used is sparse categorical cross-entropy, which is standard for multiclass classification problems. In the next cell, the author set up training logs for the TensorBoard and some TensorBoard callbacks. TensorBoard - a callback that logs training data. EarlyStopping - a callback that monitors the 'loss (function)' metric, and if the loss function does not get better in the successive ten iterations, callback stops the training and restores the network with best weights up until that iteration. TF callback that sets up TensorBoard with training logs. TF callback that contains training when the best value of the validation loss function is reached. It also restores weights from the best training iteration. The authors can see that activity stopped after 18 out of 100 epochs due to the loss function metric not changing much in the previous ten epochs; after training, the authors evaluated model accuracy and found model predicts attacks with 91.2% accuracy. Apply algorithms in pairs (DoS, BENIGN), (DDoS, BENIGN) (PortScan, BENIGN); Creation of Train and Test datasets for all data frames. First, the authors created data frames by taking rows with specific Labels from the existing database. Creation and apply algorithms for the data frame with DoS and BENIGN. Accuracy is defined in table 1,2 as the percent of positive observations properly predicted to total expected positive observations. The F1-Score is also the weighted Precision and Recall average. We got 99-percent precision and recall via Random Forest, Decision Tree, and Neural Network techniques. Finally, we attain a high F1 score and a 99-percent accuracy rate.

Table 1. Classification Report with DoS and BENIGN

Algorithms	Precision	Recall	F1-score	Accuracy
RF	0.9993	0.9993	0.9993	0.9993
DT	0.9996	0.9996	0.9996	0.9996
NN	0.9949	0.9949	0.9949	0.9949

Table 2. Confusion Matrix with DoS and BENIGN

Algorithms	Predict		
RF	188634	92	Actual
	88	58301	
DT	188662	47	
	60	58346	
NN	187949	483	
	773	57910	

Creation and apply algorithms for data frames with DDoS and BENIGN. In table 3, accuracy is defined as the proportion of correctly predicted positive observations to the total expected positive observations. Furthermore, F1-Score is the weighted average of Precision and Recall. Precision for Random Forest, Decision Tree, and Neural Network algorithms is 99 percent, and recall is 99 percent. Finally, we acquire an excellent score, F1-Score, and Accuracy of 99 percent.

Table 3. Classification Report with DDoS and BENIGN

Algorithms	Precision	Recall	F1-score	Accuracy
RF	0.9999	0.9999	0.9999	0.9999
DT	0.9999	0.9999	0.9999	0.9999
NN	0.9992	0.9992	0.9992	0.9992

Table 4. Confusion Matrix with DDoS and BENIGN

Algorithms	Predict		
RF	188719	12	Actual
	3	38395	
DT	188701	11	
	21	38396	
NN	188600	53	
	122	38354	

Creation and apply algorithms for the data frame with PortScan and BENIGN. Accuracy is defined in this table as the percent of positive observations properly predicted to total expected positive observations. The F1-Score is also the weighted Precision and Recall average. We got 99-percent precision and recall via Random Forest, Decision Tree, and Neural Network techniques. Finally, we attain a high F1 score and a 99-percent accuracy rate.

Table 5. Classification Report with PortScan and BENIGN

Algorithms	Precision	Recall	F1-score	Accuracy
RF	1.0000	1.0000	1.0000	1.0000
DT	0.9999	0.9999	0.9999	0.9999
NN	0.9954	0.9954	0.9954	0.9954

Table 6. Confusion Matrix with PortScan and BENIGN

Algorithms	Predict		
RF	188722	5	Actual
	0	27241	
DT	188715	7	
	7	27239	
NN	188226	501	
	496	26745	

Creation and apply algorithms for the data frame with Normal and Abnormal. Accuracy is defined in this table as the percent of positive observations properly predicted to total expected positive observations. The F1-Score is also the weighted Precision and Recall average. We got 99-percent precision and recall via Random Forest, Decision Tree, and Neural Network techniques. Finally, we attain a high F1 score and a 99-percent accuracy rate.

Table 7. Classification Report with Normal and Abnormal

Algorithms	Precision	Recall	F1-score	Accuracy
RF	0.9995	0.9995	0.9995	0.9995
DT	0.9996	0.9996	0.9996	0.9996
NN	0.9920	0.9920	0.9920	0.9920

Table 8. Confusion Matrix with Normal and Abnormal

Algorithms	Predict		
RF	123980	88	Actual
	65	188635	
DT	123984	77	
	61	188646	
NN	122989	1456	
	1056	187267	

Tagging attack traffic as abnormal and creating the data frame by labeling the BENIGN traffic as Normal. The author completed the CSVs of the datasets and designed them to use in KNIME. Therefore, all accuracy is based on an understanding and standard of the purport of three algorithms in the below table.

Table 9. Confusion Matrix of Random Forest, Decision Tree, k-nearest neighbors, ANN

Algorithms	DoS accuracy	DDoS accuracy	PortScan accuracy	Normal/Abnormal accuracy
RF	0.9993	0.9999	1.0000	0.9995
DT	0.9996	0.9999	0.9999	0.9996
KNN	0.9949	0.9992	0.9954	0.9920
ANN	0.7636	0.8307	0.8738	0.6034

Random Forest, Decision Tree, k-nearest neighbors have the best accuracy among the tables, giving better execution. Partition of the dataset into isolated segments like the preparation dataset is 70%, and for testing, it is 30%. This work is the proposed store-up AI method for examination attack, in which the authors can discover in the tables that the proposed methodology is showing up with 99% precision. Currently, only 79 attack insurance features are available. Later on, the authors will try all features and achieve the best accuracy. Using Random Forest returns a segment importance grid that may be will not pick parts. This framework is dreary to find information that is low and continuously exact.

5 Preprocessing & Data Cleaning

The CICIDS2017 dataset[19] used in this research is harder for a classifier, as the dataset is more resilient and contains more attack types. This is a problem for a classifier (59 percent and 55 percent more than the UNSW-NB 15 dataset, respectively). This research article is constructed such that it contains a brief overview of some existing research into cyber security deep learning applications. To illustrate design decisions of the proposed model's custom training and testing data sets, the CICIDS2017 database is evaluated during the model design phase. The CNN, a profound learning model on which the suggested IDS is based, is architecture and mathematics. The proposed parameter and IDS architecture are finally provided. To perform validation and comparison, this model was evaluated against existing benchmarks in the simulation results section. Nevertheless, in [17] this study revealed a "new hierarchical IDS based on Decision Tree and Rules-based models. They used CICIDS2017 as a dataset to evaluate the performance of their model. The proposed model combines the first stage Reduced Error Pruning Tree (REP Tree) and JRip". To classify traffic as malicious or benign, the input properties of the data set are used as input. In addition, to get the result of classification, a Forest PA Classifier employs the input features of the first dataset in the first step. Their concept has been successful in virtually every traffic class at CICIDS2017. Furthermore, to confirm the classification capacity of the authors, their proposal model was compared with 11 well-known classifications in this study. Their model had the lowest false alarm for benign transport, outperforming all other 12-classifier models in seven attack types. The overall performance of this model for the CICIDS2017 classification is competitive. As a consequence, the proposed IDS model was compared to their unique hierarchical IDS in the outcome phase of the research to evaluate the performance of the proposed model.

6 Conclusion & Future Work

This research aims to model a neural network classifier that can predict 14 network/web attacks and regular traffic with 91% accuracy. This model is proof of the concept that a feedforward neural network with a backpropagation algorithm can be used for classifying attacks in anomaly-based intrusion detection systems. The authors propose a couple of solutions for improving model accuracy. Feature engineering and feature selection can probably improve the accuracy of this model. Picking the features that have the most influence on the model. Regarding this model, the authors propose tuning the hyper-model parameters. Changing the hidden layer activation function, early stopping callback, dropout, optimizer, and loss function should increase accuracy to some extent. Another way, albeit more complicated and resource-intensive, to develop the optimum neural network design for this particular job, apply a genetic algorithm. Finally, the authors propose the usage of some other ML algorithms. Moreover, the following section includes an analysis of ML algorithms such as Random Forest, Decision Tree, k-nearest neighbors, and ANN state. Here Random Forest and Decision Tree give the most optimization accuracy. Random forest classifiers have been used in intrusion detection systems for a while now. Alternatively, the authors found some sources using autoencoders for detection.

References

1. Ho, S., Al Jurf Out, S., Dajani, K., & Mozumdar, M. (2021). A Novel Intrusion Detection Model for Detecting Known and Innovative Cyberattacks Using Convolutional Neural Networks. IEEE Open Journal of the Computer Society, 2, 14-25.
2. Samson Ho, Saleh Al Jufout, Khalil Dajani, Mohammad Mozumdar. "A Novel Intrusion Detection Model for Detecting Known and Innovative Cyberattacks Using Convolutional Neural Network", IEEE Open Journal of the Computer Society, 2021
3. Anderson, J. P. (1980). Computer security threat monitoring and surveillance. Technical Report, James P. Anderson Company.
4. Lee, W., Stolfo, S. J., Chan, P. K., Eskin, E., Fan, W., Miller, M., ... & Zhang, J. (2001, June). Realtime data mining-based intrusion detection. In Proceedings DARPA Information Survivability Conference and Exposition II. DISCEX'01 (Vol. 1, pp. 89-100). IEEE.
5. Jyothsna, V. V. R. P. V., Prasad, R., & Prasad, K. M. (2011). A review of anomaly-based intrusion detection systems. International Journal of Computer Applications, 28(7), 26-35.
6. Ciaburro, G., & Venkateswaran, B. (2017). Neural Networks with R: Smart models using CNN, RNN, deep learning, and artificial intelligence principles. Packt Publishing Ltd.
7. Kiefer, J., & Wolfowitz, J. (1952). Stochastic estimation of the maximum of a regression function. The Annals of Mathematical Statistics, 23(3), 462-466.
8. Bottou, L., Curtis, F. E., & Nocedal, J. (2018). Optimization methods for large-scale machine learning. Siam Review, 60(2), 223-311.
9. Roopak, M., Tian, G. Y., & Chambers, J. (2019, January). Deep learning models for cyber security in IoT networks. In 2019 IEEE 9th annual computing and communication workshop and conference (CCWC) (pp. 0452-0457). IEEE
10. Yeom, S., & Kim, K. Detail Analysis on Machine Learning-based Malicious Network Traffic Classification.
11. Kim, J., Shin, Y., & Choi, E. (2019). An intrusion detection model based on a convolutional neural network. Journal of Multimedia Information System, 6(4), 165-172.
12. Moustafa, N., & Slay, J. (2016). The UNSW-NB15 data set description. Unsw. ADFA. edu. au.[Online]. Available: https://www.unsw.adfa.edu.au/unswcanberracyber/cyber security/ADFANB15-Datasets/.[Accessed: 10-May-2021]
13. Khraisat, A., Gondal, I., Vamplew, P., & Kamruzzaman, J. (2019). Survey of intrusion detection systems: techniques, datasets, and challenges. Cyber security, 2(1), 1-22.
14. Ho, S., Al Jurf Out, S., Dajani, K., & Mozumdar, M. (2021). A Novel Intrusion Detection Model for Detecting Known and Innovative Cyberattacks Using Convolutional Neural Networks. IEEE Open Journal of the Computer Society, 2, 14-25.
15. Gu, Y., Li, K., Guo, Z., & Wang, Y. (2019). Semi-supervised k-means DDoS detection method using hybrid feature selection algorithm. IEEE Access, 7, 64351-64365.
16. Atefi, K., Hashim, H., & Kassim, M. (2019, December). Anomaly analysis for the classification purpose of the intrusion detection system with K-nearest neighbors and deep neural network. In 2019 IEEE 7th Conference on Systems, Process and Control (ICSPC) (pp. 269-274). IEEE.

17. Ahmim, A., Maglaras, L., Ferrag, M. A., Derdour, M., & Janicke, H. (2019, May). A novel hierarchical intrusion detection system based on decision trees and rules-based models. In 2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS) (pp. 228-233). IEEE.
18. Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980
19. Tian, Xuefei, et al. "IntruDTS: Interactive Visual Analysis System for Intrusion Detection in Time Series." 2020 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom). IEEE, 2020.
20. Samson Ho, Saleh Al Jufout, Khalil Dajani, Mohammad Mozumdar. "A Novel Intrusion Detection Model for Detecting Known and Innovative Cyberattacks Using Convolutional Neural Network", IEEE Open Journal of the Computer Society, 2021
21. S. F. Shetu, M. Saifuzzaman, N. N. Moon and F. N. Nur, "A Survey of Botnet in Cyber Security," 2019 2nd International Conference on Intelligent Communication and Computational Techniques (ICCT), 2019, pp. 174-177, doi: 10.1109/ICCT46177.2019.8969048.
22. Intrusion Detection Evaluation Dataset, accessed May 18, 2021, from unb.ca/cic/datasets/ids-2017.