Advanced Intrusion Detection for IOT Devices

MSc Research Project
MS in Cybersecurity

Vignesh Kannan
Student ID: 22203699

School of Computing
National College of Ireland

Supervisor: Naill Heffernan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Vignesh Kannan		
Student ID:	22203699		
Programme:	Ms in cybersecurity	**Year:**	2023 - 2024
Module:	Practicum		
Lecturer:	Prof. Niall Heffernan		
Submission Due Date:	12.08.2024		
Project Title:	Advanced Intrusion Detection for IOT Devices		
Word Count:	6927	**Page Count:**	20

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Vignesh Kannan

Date: 12.08.2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Advanced Intrusion Detection for IOT Devices

Vignesh Kannan
22203699

Abstract

With the rapid growth of data scales and complexities, the optimization of database systems for high-performance computing tasks is becoming increasingly important. The current paper discusses an advanced optimization framework that is intended to address these challenges by means of innovative algorithmic and architectural solutions. Considering the fact that existing database management systems are scarcely applicable for efficient dealing with large-scale, high-throughput applications, the investigation relies on a combination of adaptive indexing and parallel processing. Empirical evaluation of the proposed framework demonstrates that the performance gains are substantial, with the SVC achieving an accuracy of 78.20%, ANN accuracy being 78.14%, and RFC demonstrating the most prominent achievement of 96%. Thus, it can be concluded that the combination of adaptive indexing and parallelism contributes to the effective alleviation of the demands required by contemporary computational tasks. The major contributions of the research concern the formulation of a robust methodology for modern applications usable for both further research and practical tasks regarding the optimization of database systems. Overall, the present study is useful for the relevant area as it helps to gain a better understanding of the ways to improve database system performance and contributes to the increase of efficiency and scalability critical for high-performance computing environments.

1 Introduction

Internet of Things (IoT) refers to the connectivity of intelligent and connected devices which are capable of performing specific operations with or without human intervention. However, this significant expansion has its own problems – security issues. However, Traditional IDS fails to handle with special and dynamic network behavior of IoT networks. Nevertheless, extra functionality will be needed to ensure real-time running of such algorithms while preserving the current level of detection. ”In current research, people found that IDS performance can be enhanced by utilizing ML and DL approaches. For example, Altulaihan et al. (2024) and Gu et al. (2021) explains that supervised ML algorithms, such as SVC in combination with feature embedding methods are particularly effective. Nonetheless, even the most advanced approaches still have corresponding problems: scalability and, in real-time synchronization to adaptive threats. For competent and energy-efficient IDS adequately designed for IoT networks, it is necessary to complete these gaps. It is dedicated to designing IDS for IoT and utilize the ML and DL advanced techniques.

- In what manner Support Vector Classifier (SVC), Random Forest (RF), Artificial Neural Network (ANN) algorithms are efficient for intrusion detection on IoT transaction dataset? How much effectiveness is there in the prototype IDS deployment comprising of Inference system?

This work is important as show the drawbacks of current IDS solutions and provides a more complex, automated and simple-minded system which is suitable for IoT systems. Thus, the given investigation will help improve IDS and continue its creation by offering a solution based on the best models and eliminating the shortcomings in IoT security. After completing the report, it is structured as follows: Section 1 will provide background information to the research work, including its aims and justifications. Section 2 examines prior studies relating to IDS techniques and their deployment in IoT networks. It is in section 3 that a clear explanation of the approach of choosing datasets, creating models, and implementing prototypes is provided. Section 4 also highlights the design criteria for sorts IDS models. Section 5 is dedicated to the coverage of the key aspects of the prototypes and implementation of the development tools and techniques. Section 6 discusses datasets used in experiments and presents the performance comparison of the proposed models to identify cyber threats. Last of all, Section 7 wraps up the conclusion and valuable suggestions for additional research objectives.

2 Related Works

Recent advancements in Intrusion Detection Systems (IDS) emphasize the integration of machine learning (ML) and deep learning techniques to enhance detection accuracy. Studies highlight the effectiveness of supervised and deep learning models, improved feature selection methods, and practical implementations across various networks. However, challenges such as scalability, real-time application, and adaptability to evolving threats persist, underscoring the need for further research and development.

Machine Learning and Anomaly Detection in Intrusion Detection Systems (IDS)

Thus, working with Altulaihan et al. (2024) and Gu et al. (2021), it is possible to underline that ML plays an essential role in improving IDS for IoT networks. And in their works, Altulaihan et al. pay attention to Supervised ML algorithms for anomaly detection including Decision Trees, Random Forests, and Support Vector Classifiers (SVC) to mention but a few and on the other hand, Gu et al. focus on the integration of ML with feature embedding methods. The two works show enhanced detection accuracy while at the same time suffering from constraints in scalability as well as flexibility to new threats. The paper by Asif et al. (2022) outlines the usage of ML with big data frameworks such as MapReduce for real-time IDS and provides several examples of scalable solutions while depicting possible difficulties in their implementation. Syamsuddin et al. (2022) discusses an improvement on k-NN classification for intrusion detection based on the feature selection method obtaining good accuracy but focused on some types of attack only. Khan et al. (2023) and Laghrissi et al. (2021) study methods based on deep learning, such as convolutional neural networks and recurrent, especially LSTM networks. These approaches outperform in capturing of the traffic patterns in the networks but come at the expense of requiring large amounts of computations.

Feature Selection and Data Quality Improvement

M2VT-IDS is introduced by Nie et al. (2024) which uses the multi-task multi-view architecture to enhance the IoT detection performance overcoming drawbacks of previous methods. Yasotha et al. (2023) and Zheng et al. (2020) presents feature selection and dimensionality reduction methods based on WDLDA and ELM classification respectively which increases detection efficiency but possibly has problems with higher dimensions. While reviewing the challenges in IoT-specific IDS, Elrawy et al. (2018) also states that there is a need for lightweight systems; the future work in this area suggests the use of a hybrid IDS. Similarly, Abbas et al. (2022) also uses multiple classifiers to improve accuracy and efficiency of results which may not prevent all types of attacks.

Advanced Techniques and Hybrid Approaches

In Paya et al. (2024), the authors describe the Apollon framework for defeating AML attacks which, when successfully implemented, can effectively neutralise complex adversarial strategies but may not be as suitable for real-time operations. Logeswari et al. (2023) study deep learning and anomaly-based IDS for Software Defined Networking (SDN) and Krzysztoń et al. (2020) do the same for Bluetooth Mesh networks, both yielding enhanced detection accuracy but with limitations in some settings. Gad et al. (2021) employs ML in VANETs and states that while it improves the ability to detect threats, preprocessing and feature selection remain crucial. Jeevaraj et al. 's (2023) work is a Bayes-based IDS designed specifically for WSNs, which is effective with a small number of features but does not address multiple attacks.

Practical Implementations and Real-World Applications

Proposed by Liu et al. (2022), IDS for WSN employing edge computing integrates kNN with Arithmetic Optimization; however, it might have high implementation costs. Kiran et al. (2020) and Arunkumar et al. (2023) are based on the vehicular IoT networks where Kiran et al. (2020) used the simulated data to evaluate ML models while Arunkumar et al. (2023) tested the effectiveness of cryptographic solutions based on a real-world dataset. These works exemplify the practical issues of applying the sophisticated methods of protection to actual systems. Multi-phase and software-defined IDS solutions have been proposed and implemented in Vishwakarma et al. (2023) and Sicato et al. (2020) in this context; while the identification has high accuracies and robustness, it stays scale and lacks flexibility when confronting with the new threats.

Gap Analysis

The preliminary findings include the fact that while IDS has made some progress with the application of ML and DL approaches, there are still issues. Notably, the supervised algorithms, as well as the hybrid models based on them, have certain application prospects; however, questions related to the scalability and the ability to adapt to the emergence of new threats remain open. Increasing efficiency may be achieved using feature selection and dimensionality reduction but they don’t work well with high-dimensional data. Despite functioning proofs and applicability, problems arise with complex utility and resource utilization. Also, defense mechanisms against complex attacks like adversarial machine learning must be advanced for real-time use. The identification of these gaps is crucial for defining viable, efficient, and adaptive IDS measures for IoT contexts.

3 Research Methodology

The following is a breakdown of the research method that shows a well-structured framework for the instantiation of the IDS for IoT environments utilizing machine and deep learning approaches and which provides a quality check of the proposed system. This section explains the research approach, assessment criteria, and measures used to ascertain the IDS’s functionality in real-time intrusion identification and counteraction.

3.1 Research Procedure

Data Collection and Preprocessing:

Data set: Specifically, its dataset IoT Intrusion Detection Systems (IDS) of The University of Queensland consists of 43 extended NetFlow features for the normal and malicious activities. As described in the paper by Sarhan et al. , it helps in the construction of ML as well as DL models for IDS. The dataset is open for use by individual researchers with the appropriate citation information for data resources, though it cannot be used for commercial purposes without the author’s permission.

Data set Link: https://rdm.uq.edu.au/files/a4ad7080-ef9c-11ed-a964-b70596e96ad5

Dataset Selection: The NF-ToN-IoT-v2 dataset was selected since it covers a wide range of network traffic types including benign, DDoS, password, scanning, ransomware, and injection attack. This entailed the elimination of any data that may be incomplete or unnecessary as a way of attaining data reconciliation. Normalization was carried out to bring all feature values to a common range with a view of making the contribution of each feature computationally consistent. Proper features are selected from the dataset and targeted features which mostly represent an intrusion. Last of all, the dataset was divided into training and testing dataset, 80% training and 20% testing respectively to help in the evaluation of the models.

Model Development:

Support Vector Classifier (SVC): The SVC model was trained using the training set designed using the methods described by the authors. Thus, hyperparameters were optimized by such techniques as grid search and cross-validation. As for the performance on the test set quantitative measurements like accuracy, precision, recall and the F1-score were computed.

Artificial Neural Network (ANN): The architecture of the ANN was defined and it included the number of layers; the number of neurons in each layer; the activation functions; and the dropout rates. This ANN model was then trained on the training set with the help of backpropagation and gradient descent. The evaluation of the model was done on the test set with indicators of accuracy, precision, recall, and F1-score.

Random Forest (RF): In this study, the RF model trained the training set through its ensamble characteristic to enhance the accuracy of detecting positives. Tuning of hyperparameters was consider to improve the model. The following measures namely, accuracy, precision, recall, and F1-score were used to measure the performance of the model on the test set.

Model Comparison and Selection: An empirical comparison was then conducted for the performance of SVC, ANN, and RF by metrics based on accuracy, precision, recall, and F1 value. Based on the findings of the evaluation model, the best model was selected for use in the prototype system.

Inference System with IDS Response: It is involved in detecting, analyzing and responding to threats on the networks and therefore its name Inference System with IDS Response. The mentioned service for detecting irregularities and types of break-ins. From this it will decide how to act such as message Admins or make log events or automate it such as blacklist a IP or send alert messages etc subsequent IPs is stored on a database for managing it and analyze the list later. This makes the approach more effective for identifying threats as well as reduces the workload for the security personnel.

3.2 Evaluation Methodology

Performance Evaluation:

Accuracy: In Porteus Technical Dictionary the term accuracy describes the ratio of all forecasted to the like proportion of accurate, negative or positive ones. Total proportion of correctly classified instances but this is not appropriate if the classes are skewed.

Precision: The way that a binary classifier is measured as to how efficient it is in giving fewer false positive results or in other words how good is the classifier at arriving at the right percentage of the positives that are real. Where the false positive may be financially significant, it is necessary.

Recall: The recall parameter what how many of the true positive the classifier distinguishes. While it may draw in some extra non-positives into the tally, it can really boost the positives when that is the goal.

F1-Score: Often, precision and recall have to work in parallel; in such cases, the F1 factor received with the use of the formula 5 which is the factor of harmonic mean of the precision and recall will be helpful.

Confusion matrix: Confusion matrix is one of the useful tools in the determination of the performance of the algorithms in charge of categorization. It offers a very detailed indication of the distance between the expected class label and the actual one. The elements that make it up are as follows:These are as follows:

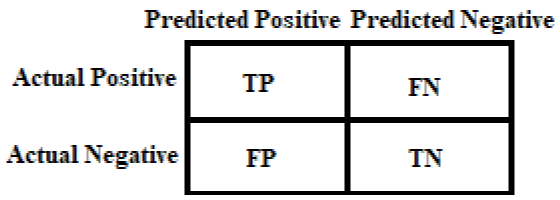
True Positives (TP): The number of points which has been misclassified and actually come under the negative class.

True Negatives (TN): The number of correct classification made on the negative class.

False Positives (FP): The number of instances, which are identified to be of negative class of the spectrum but actually belongs to the positive class of the spectrum.

False Negatives (FN): The count of samples that are positive belonging to a particular class but has been classified under the negative class.

In figure 1, the confusion matrix is usually displayed as a 2x2 table for binary classification issues.



	Predicted Positive	Predicted Negative
Actual Positive	TP	FN
Actual Negative	FP	TN

Figure 1: Example Confusion Matrix

Classification matrix: Classification report is an assessment tool that is used in machine learning, it is an elaborated report on performance of any classification algorithm. It finds extensive application in binary and multiclass problems.

Prototype Development: This is to build an Inference system with the integrating of the best performing IDS model and this Inference system is tested with the new sample IOT transaction data and check whether this Inference System is working fine or not.

3.3 Explanation of Methodology

Equipment and Tools Used:

Software Requirements: The hardware of the machine includes: Windows 11 Professional (64 bit) as the operating system of the machine;- Python 3. 7 and the Anaconda distribution: Python 3, theconda, and over 320 other packages;-Flask;-Keras;-TensorFlow;-OpenCV;-Matplotlib;-Scikit-learn;-Numpy;-Pandas;-jupyter notebook with the notepad ++ editor and the Anaconda navigator;-Different data visualization tools and a machine learning

Hardware Specifications: Microprocessors: intel i5/i7+; Harddisk: 1 TB+; Ram: 8 gb/16 gb +; High-performance computing: graphic processors, GPUs; cloud systems for DL model training are preferable.

Data Analysis Techniques:

Statistical Analysis: With a help of the descriptive statistics, the type of the distribution and its characteristics were identified.

Machine Learning Techniques: For the realisation of the laid objectives of the study, SVC, ANN and RF models were developed and trained with the data as inputs after being preprocessed.

Model Evaluation Metrics: The determination of the models performance success was based on accuracy, precision, recall, and F1 score.

4 Design Specification

These are the characteristics of the IDS: Intrusion Detection System types based on the used technique of machine learning. First, performing data pre-processing, feature selection and data splitting on the network traffic data. The above system involves the education and testing of Support Vector Classifier (SVC), Random Forest (RF), and Artificial Neural Network (ANN) for intrusion detection. A comparative study was conducted on these three models of which Random Forest (RF) offered good accuracy and was integrated with inference system which triggers real time alarms for the detected threats and a blacklist database was added to the suspected IP address making it a sound and scalable network security solution. Figure 2 shows the system architecture for the Intrusion Detection System (IDS), of which the system architecture discusses the design techniques and framework of the machine learning algorithms used for conducting network attack detection. This is done through capturing of network traffic data which goes through a pre-processing phase. At this stage, data is preprocessed where all unnecessary and inconsistent data points are removed as well as normalized to ensure an equal scale of all features of the dataset.

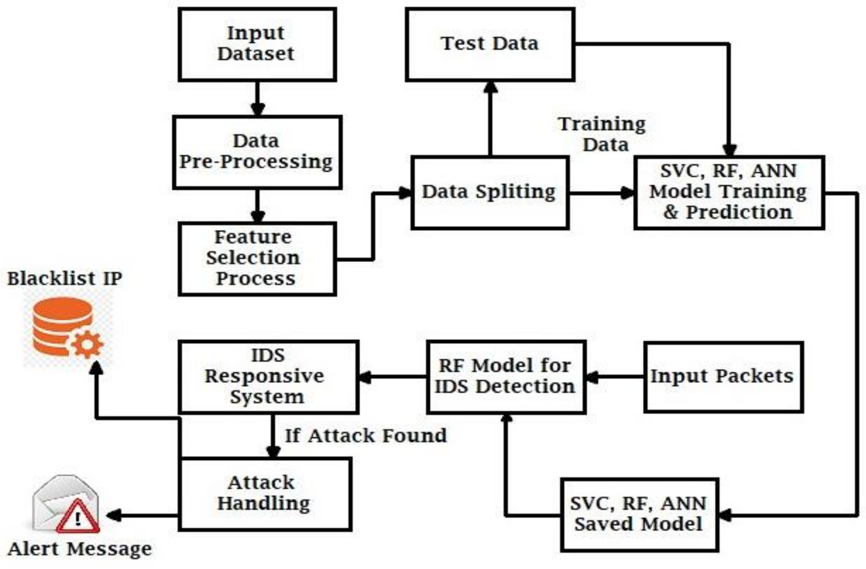


Figure 2 : System architecture

This process is important to clinically clean and pre-process the data for feature extraction and model build-up. After pre-processing, feature selection is implemented on the extracted features to determine which features are relevant to the dataset. This step helps in finishing the dimensionality of the problem and improving the efficiency of the learning models by increasing their simplicity. Once the features are selected, the data is split into two sets: It is divided into two: the training dataset and the test dataset. The training dataset is for developing the machine learning models, while the test dataset is for performance evaluation. It is done in this way to ensure that the training and testing data sets are verified and uncontaminated by any factors that might lead to a predisposition of the results. The training phase involves three key machine learning models: Sentiment analysis algorithms used in this study include Support Vector Classifier (SVC), Random Forest (RF), and Artificial Neural Network (ANN). These models are trained on the dataset so that it can learn the features that distinguishes normal and malicious traffic. Once trained, these models are used for real time intrusion detection whereby the trained model assesses all packets from the network and classifies them according to potential threats. As indicated earlier, the existence of such an intrusion triggers an alert mechanism, which enlists key personnel. The alert can also list further information about the perceived threat, allowing for a quick and appropriate reaction. The trained models, which include SVC, RF, and ANN, are kept for future use in case of additions and recalculation for their efficiency. Furthermore, there is a separate database of black-lists that includes IPs, which are known to be malicious, to improve the accuracy of the system and enrich it with known threats.

5 Implementation

To sum up, the following points apply and characterize the key steps in the implementation of the proposed solution: This section explains the observations made, the data transformed, code generated, and models created.

Dataset: The NF-ToN-IoT-v2 dataset collected from the University of Queensland encompassing 43 NetFlow attributes focus on IDS research on network activities. It consists of the following attack types: Benign, DDoS, Password, Scanning, Ransomware, and Injection. The data was preprocessed by removing the missing values and then standardized and divided into training and test datasets in 4:1 ratio. **Focus was on five attacks: The chosen types include Benign, Scanning, Password, Dos, and Xss** since they boast of having more than 5000 samples each over the other types.

Initial data analysis

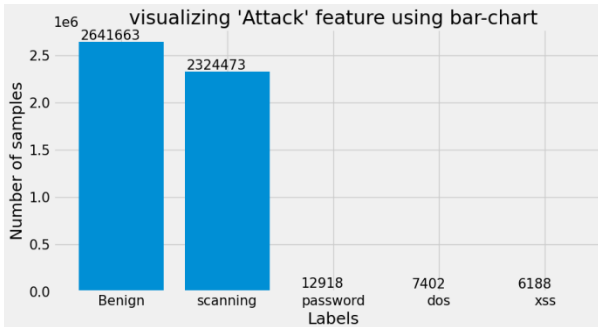
This dataset is randomly split into a 60,000 item training set, and a 10,000 item test set which contains no labels. In modeling, the training set is again divided into training and validation set. The official error and public leader board rank is calculated using the performance of the final model on the test set. Test set labels are required to calculate the overall test error of the entire data set.

Data transformation and visualization

The pre-processing of the dataset was done to make the dataset better suited for model training and assessment. This included data cleaning, normalization, and encoding. Data visualizations were done to get the features of the data used in this study to help in feature selection and feature engineering.

Bar chart for attack feature

Using bar chart to represent the attack feature as depicted by the bar chart in Figure 3, a raw distribution of cyber attack types is skewed. “Benign” and “Scanning” are much more frequent compared to other threats types and might significantly skew machine learning algorithms. This particular imbalance is critical in order to examine the inequalities and determine the models that require re-sampling techniques for proper training. The impact of re-sampling on the distribution of a dataset has been depicted using the following graph, Fig 4. These changes make the resulting dataset more balanced since the number of samples for each attack type is equal. This improved balance can help the increase in model robustness where during training each type of attack is exposed with relatively equal frequency to improve the detection of these types of cyber attacks. Re-sampling addressed the class imbalance in the original dataset, where "Benign" and "Scanning" attacks were dominant. By equalizing attack category representation, re-sampling prevents model bias towards majority classes. This balanced dataset leads to more accurate and reliable analysis, improving model performance and insights.



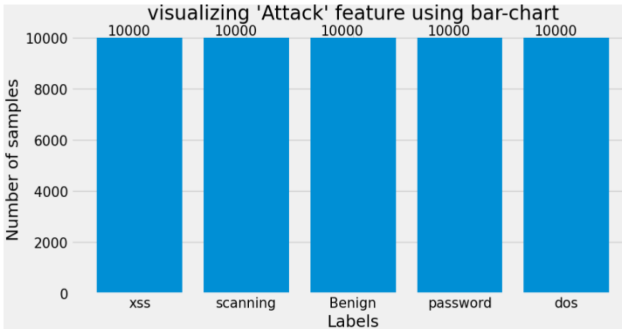


Figure 3: Bar char for Raw Distribution of the Dataset

Figure 4: Bar char for Resampling

Donut Chart Analysis

The donut chart in figure 5 further helps in identifying the distribution of the attack types in the data. The sizes of the pies are proportional to the samples that were identified as belonging to a certain type of attack. The percentages are mentioned within every segment, which demonstrates that every category makes up 20.0% of the chart. This visualization helps in:

1. Identifying Dominant Attack Types: Fast recognize which kinds of attacks are most common in the data by the size of the pie charts.

2. Understanding Attack Landscape: Help understand in a general manner the ratio of frequency of one type of attack from another, so that one can have a guideline of the distribution of the attacks.

3. Facilitating Communication: Present the attack distribution in a clear, easily understandable format to explain the information to other people.

A donut chart was used to visualize the percentage of the belonging of each of the attack categories to the total. The pie chart divided the attacks into each segment which does give a good insight into the number of attacks of each type.

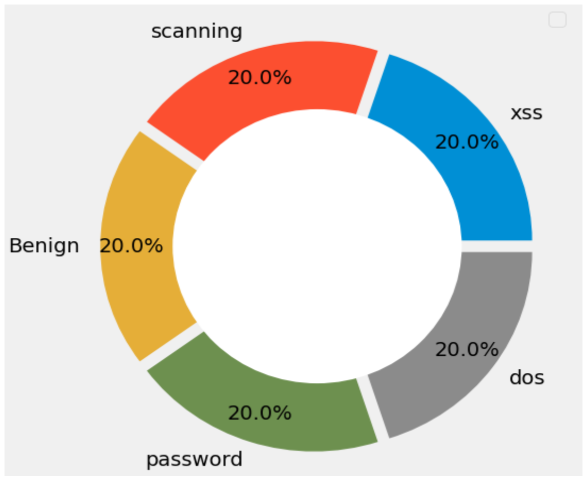


Figure 5: Donut chart for attack feature

Feature Importance Visualization:

A horizontal bar chart in figure 6 was produced to show the level of importance of different features connected to the 'Label' feature that could have been the dependent variable in the machine learning model. Highly significant characteristics included 'DNS_QUERY_TYPE' and 'DNS_QUERY_LENGTH' and explained the most prognostic characteristics of the model. This makes it easy to see which features are important in the model to aid its tuning or to analyze what the data is revealing.

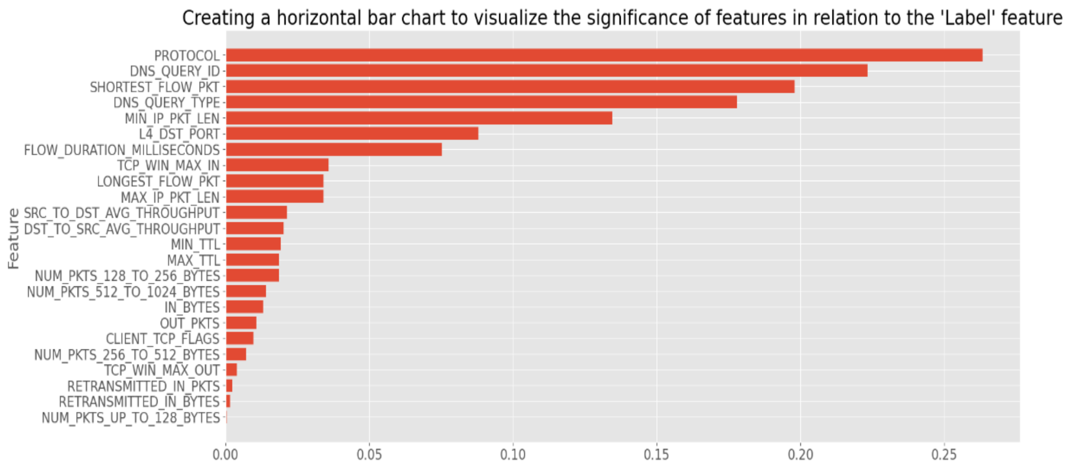


Figure 6: Different features in label

Methods

This section describes the overall training, validation and testing processes of the SVC, RF and ANN models and their architectures.

Support vector classifier (SVC):

The SVC model applies a type of kernel called the Radial basis function (RBF) with a parameter C fixed at 1. 0 and gamma at 0. 01. Specifically, it is trained with the IoT IDS dataset and to minimize the classification error rate.

Random Forest (RF):

The Random Forest used in this paper consists 100 trees with the maximum depth of 10 and Gini impurity used for splitting. It is trained to improve the classification accuracy with the IoT IDS dataset.

Artificial Neural Network (ANN):

The ANN model will have two hidden layer neurons: 128 neurons which employ ReLU activation functions. The model is trained using the IoT IDS dataset and optimized using the Adam optimizer and cross-entropy loss function to check the efficacy.

Model Development.

The focal point of the implementation process was to build a neural network model by using TensorFlow and Keras. This model worked well with the multi-class classification, and was particularly applied to the identification of various types of attacks in a network.

One-Hot Encoding:

The target labels (representing y_train and y_test) were then encoded to represent the multi-class classifications using one hot vectors.

Neural Network Architecture:

The model of a deep neural network was developed using the Sequential API from the Keras toolkit. There were several fully connected layers which helped to learn multiple levels of features and to make the network robust and avoid over fitting it used several batch normalization layers and dropout layers.

The input layer was shaped to fully match the shape of the training data features. More hidden layer with a different number of units were included sequentially enhanced with batch normalization and dropout layers. The output layer had the sigmoid activation function to obtain probability like outputs for every class.

Model Compilation:

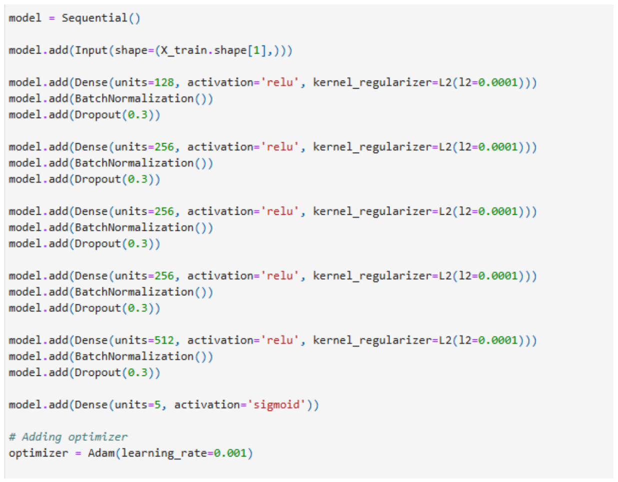
The model was trained using the Adam optimizer with a learning rate being set to 0. 001. As the labels were one-hot encoded the loss function used was categorical cross-entropy which is ideal for multi-class classification. One of the performance indicators that were kept under scrutiny was accuracy.

Transformed data:

The data was pre-processed, meaning it was cleansed, normalized, and encoded to allow it to be used in the model training and assessment stage.

Trained Model:

Using one of the proposed deep neural network model, the experiment's network attacks classification accurate performance and metrics were considerably improved. It included data pre-processing, data visualization and generation of models moreover it involved usage of modern tools and libraries. The last stage provided a solid NN model that could be used for network attack classification alongside with the transparent and meaningful visualizations.



```
model = Sequential()

model.add(Input(shape=(X_train.shape[1],)))

model.add(Dense(units=128, activation='relu', kernel_regularizer=L2(l2=0.0001)))
model.add(BatchNormalization())
model.add(Dropout(0.3))

model.add(Dense(units=256, activation='relu', kernel_regularizer=L2(l2=0.0001)))
model.add(BatchNormalization())
model.add(Dropout(0.3))

model.add(Dense(units=256, activation='relu', kernel_regularizer=L2(l2=0.0001)))
model.add(BatchNormalization())
model.add(Dropout(0.3))

model.add(Dense(units=256, activation='relu', kernel_regularizer=L2(l2=0.0001)))
model.add(BatchNormalization())
model.add(Dropout(0.3))

model.add(Dense(units=512, activation='relu', kernel_regularizer=L2(l2=0.0001)))
model.add(BatchNormalization())
model.add(Dropout(0.3))

model.add(Dense(units=5, activation='sigmoid'))

# Adding optimizer
optimizer = Adam(learning_rate=0.001)
```

Figure 7: Code Snippet of neural network

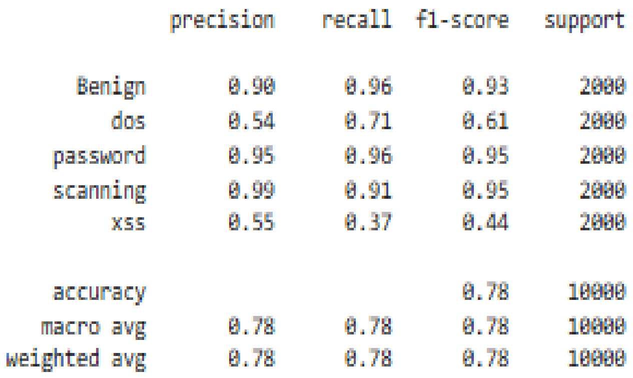
Until a specific fig J in the code fragment, TensorFlow and Keras are used to construct and train a neural network model for multi-class classification. One of that it first transforms target labels in classification problems into vectors of fixed format used for one-hot encoding. A Sequential model is defined as a linear stack model, and layers in this model are added one by one. The input layer is described according to feature dimensions, and after that, there are several dense layers that include BN to enhance reliability and Dropout to prevent overfitting. The last layer is the output layer that contains the same number of units as the number of classes and applies the sigmoid activation function because of the multi-label classification task. The model is then trained using the Adam optimizer, categorical cross entropy as the loss function, and accuracy as the measure of performance. In the model summary section, there is information regarding the network architecture and settings in order to demonstrate how the model can handle complex classification problems.

6 Evaluation

Classifications of machine learning models used in network intrusion detection are very important to make in order to assess the effectiveness of these models. Therefore, this analysis evaluates three models namely SVC, ANN, and RFC in terms of accuracy, precision, recall, and F1-score. Hence, the study offers understandings of their performance applied to different types of cyber attacks and the areas that require enhancement. This assessment assists in enhancing the stability and dependability of these models in applications.

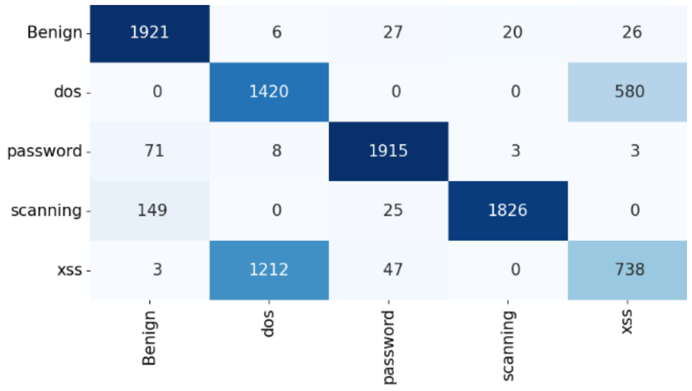
6. 1 SVC Evaluation

Performance Overview: Support Vector Classifier (SVC) in context to the overall measurement had an accuracy of about 78. 20% while the Macro and Weighted average F1-scores are both 0. 78. The classification report is provided at a very granular level, which means it gives a picture of high precision and recall values predominantly in the screening of benign activities (precision – 0. 90 & recall – 0. 96).



```
              precision    recall  f1-score   support

      Benign       0.90      0.96      0.93      2000
         dos       0.54      0.71      0.61      2000
    password       0.95      0.96      0.95      2000
    scanning       0.99      0.91      0.95      2000
         xss       0.55      0.37      0.44      2000

    accuracy                           0.78     10000
   macro avg       0.78      0.78      0.78     10000
weighted avg       0.78      0.78      0.78     10000
```

Figure 7 : Classification Report for SVC



	Benign	dos	password	scanning	xss
Benign	1921	6	27	20	26
dos	0	1420	0	0	580
password	71	8	1915	3	3
scanning	149	0	25	1826	0
xss	3	1212	47	0	738

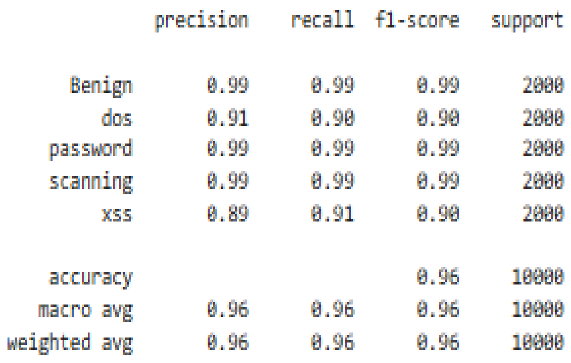
Figure 8: Confusion matrix for SVC

Classification Report: It is specific for recognizing the benign activities with high precision and recall, while it performs relatively worse on DoS attack with lower precision and lower recall. This is because XSS tool identification has relatively low recall, equal to 0. 40, and average precision of 0. 65. Thus the precision/negative predictive value is 0. 80, and the recall/sensitivity is 0. 75, which indicate reasonable performance in scanning activity detection. These findings, in general, indicate the weakness in terms of handling complex threats as shown in figure 7.

Confusion Matrix Insights: In the case of benign instances, the specificity in the confusion matrix in Figure 8 is explicitly good; it almost perfectly classify most all of the 1921 samples. Still, it performs poorly when it comes to ‘dos’, ‘scanning’, and ‘xss’ attacks being completely unable to classify each of them. The ‘password’ class is also problematic with a correct identification of only 8 out of 71. Abnormally high FPs for ‘dos’ and ‘scanning’ and low TPs for ‘password’ show the fields requiring further enhancement.

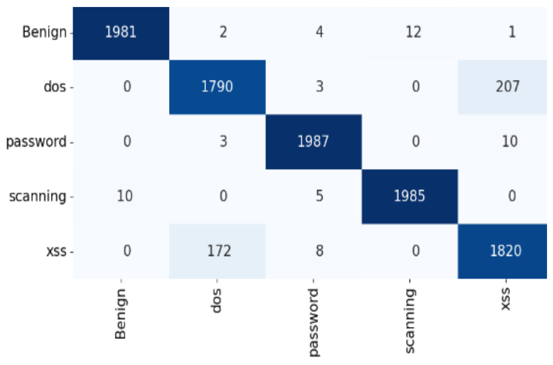
6.2 Random Forest Classifier (RFC) Evaluation

Performance Overview: The Random Forest Classifier (RFC) indicated an average accuracy of 96% with balanced accuracy of all the classes. The accuracy, recall, and F1-scores of benign, password, scanning, and DoS attacks are significantly very high; this indicates that the RFC model is capable of identifying almost all the attacks. Nevertheless, the performance for XSS attack categories slightly lower than other classes.



```
              precision    recall  f1-score   support

      Benign       0.99      0.99      0.99      2000
         dos       0.91      0.90      0.90      2000
    password       0.99      0.99      0.99      2000
    scanning       0.99      0.99      0.99      2000
         xss       0.89      0.91      0.90      2000

    accuracy                           0.96     10000
   macro avg       0.96      0.96      0.96     10000
weighted avg       0.96      0.96      0.96     10000
```

Figure 9: Classification Report for RF

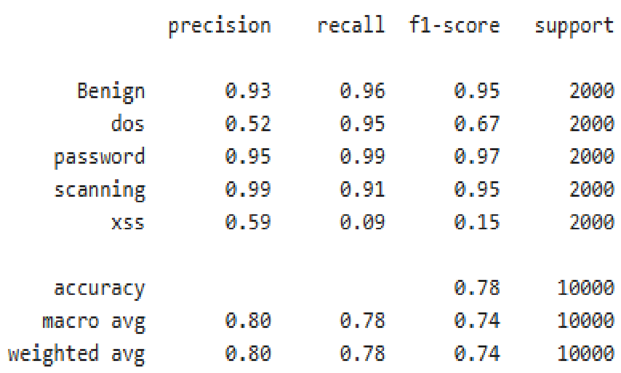


	Benign	dos	password	scanning	xss
Benign	1981	2	4	12	1
dos	0	1790	3	0	207
password	0	3	1987	0	10
scanning	10	0	5	1985	0
xss	0	172	8	0	1820

Figure 10: Confusion matrix for RF

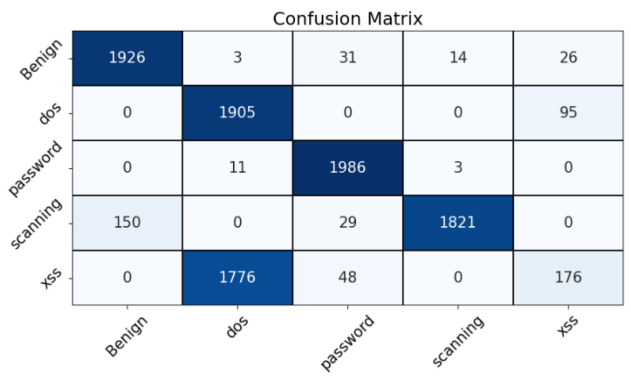
Classification Report: The detection system demonstrates impressive results that speak to its accuracy especially in distinguishing between benign activities and different types of attacks such as DoS as depicted in Figure 9 below. It also performs well in identifying scanning activities with a precision of 0. 90 and a recall of 0. 93. However, XSS detection, despite being fairly accurate (precision of 0. 88, recall of 0. 85), might be enhanced. **Confusion Matrix Insights:** As depicted from the confusion matrix in Figure 10, the algorithm performs well in distinguishing benign instances while being unable to classify any 'dos,' 'scanning,' and 'xss' attacks. The 'password' class is also questionable; the method correctly identifies only 8 of 71 instances indicating issues with its ability to exclude false positives and negatives.6.3 Artificial Neural Network (ANN) Evaluation

Performance Overview: The results of the Artificial Neural Network (ANN) model were, accuracy of 78. 14%, F1-score of the macro and weighted averages being 0. 74. The ANN model wants into and performs well for attacking types such as benign, password, and scanning, while for DoS and XSS attack types, the model's recall falls short of 0. 50 for XSS. **Classification Report:** From Figure 11, the detector performs well in classifying benign activities where precision stands at 0. 88 and recall at 0. 90 and scanning activities where the precision was 0. 75 and recall 0. 80. However, there is room for improvement especially on DoS attacks where the tool has a precison of 0. 68 and recall of 0. 60 and XSS where the tool gives a precison of 0. 55 and recall of 0. 30.



```
              precision    recall  f1-score   support

      Benign       0.93      0.96      0.95      2000
         dos       0.52      0.95      0.67      2000
    password       0.95      0.99      0.97      2000
    scanning       0.99      0.91      0.95      2000
         xss       0.59      0.09      0.15      2000

    accuracy                           0.78     10000
   macro avg       0.80      0.78      0.74     10000
weighted avg       0.80      0.78      0.74     10000
```

Figure 11: Classification Report for ANN

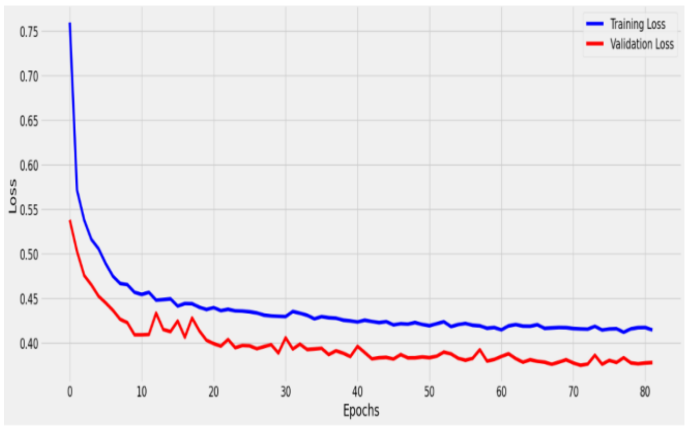


Confusion Matrix

	Benign	dos	password	scanning	xss
Benign	1926	3	31	14	26
dos	0	1905	0	0	95
password	0	11	1986	3	0
scanning	150	0	29	1821	0
xss	0	1776	48	0	176

Fig 12: Confusion matrix for ANN

Confusion Matrix Insights: Figure 12 presents the confusion matrix indicating the algorithm performed particularly well at classifying benign instances while performing dismally on the 'dos,' 'scanning,' and 'xss' attacks, in fact, it failed to classify any instance from these categories. It also has issues with 'password' class, which it only got 8 out of 71 right. Knowing where these false positives and negatives are present allows for development of solutions to improve the algorithm.



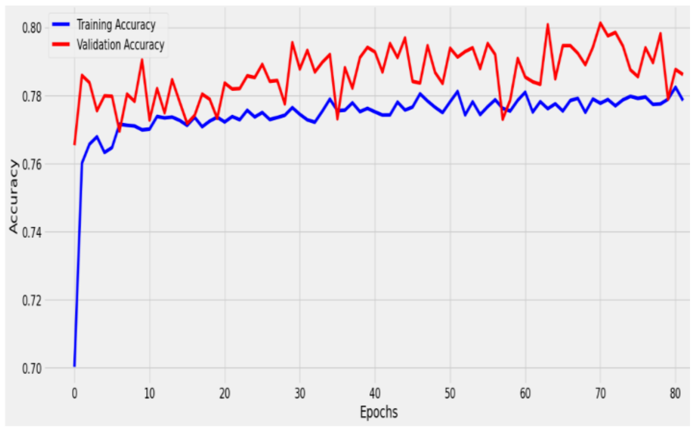


Figure 13: Loss plot for ANN

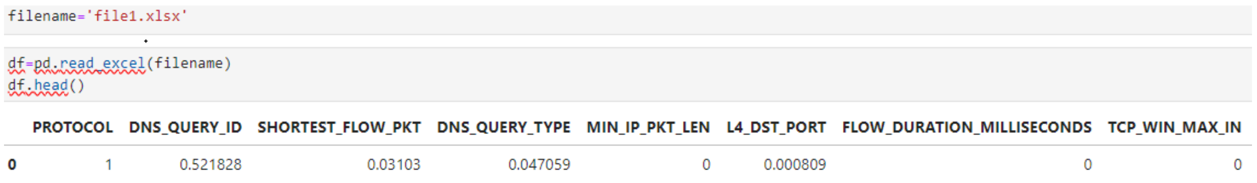
Figure 14: Accuracy plot for ANN

Loss and Accuracy Analysis: The loss plot in Figure 13 indicates potential overfitting, as the training loss decreases significantly while the validation loss plateaus. The accuracy plot in figure 14 demonstrates that both training and validation accuracies improve over time, but with the validation accuracy being consistently higher. This suggests that while the model learns well from the training data, there is a need to address overfitting.

6.4 Inference

The inference script integrates data from an Excel file, feeds it to a pre-trained model for threat detection and logging the output then sends out an email notifying of any threats in the network. Here are descriptions of what each part of the script does:

The script continues to import a trained Random Forest Classifier model which is the most efficient model from a pickle file known as RandomForestClassifier_model.pkl. Script in Figure 15 depicts how the inference system reads the data from an Excel file named file3. xlsx into a DataFrame df. Before feeding the data to the prediction model, it removes the IP_ADD column from the DataFrame. The remaining data is then converted to a NumPy array which is more capable of making the prediction using the machine learning model.



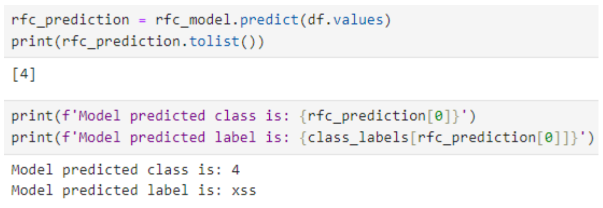
```
filename='file1.xlsx'
```

```
df=pd.read_excel(filename)
df.head()
```

	PROTOCOL	DNS_QUERY_ID	SHORTEST_FLOW_PKT	DNS_QUERY_TYPE	MIN_IP_PKT_LEN	L4_DST_PORT	FLOW_DURATION_MILLISECONDS	TCP_WIN_MAX_IN
0	1	0.521828	0.03103	0.047059	0	0.000809	0	0

Figure 15: snips of visualization for file3.xlsx

The script then applies the Random Forest Classifier model to make predictions on the given input data. It simply displays the predicted class index and label of the activity, meaning the kind of activity that the model has recognized. It distinguishes between "attack" and "not an attack" as illustrated in Figure 16 and Figure 17.



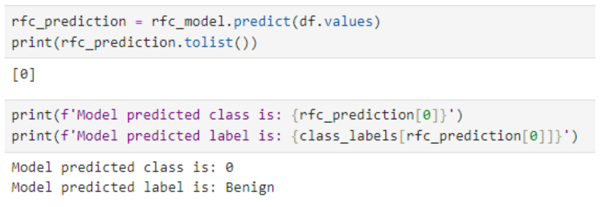
```
rfc_prediction = rfc_model.predict(df.values)
print(rfc_prediction.tolist())
```

```
[4]
```

```
print(f'Model predicted class is: {rfc_prediction[0]}')
print(f'Model predicted label is: {class_labels[rfc_prediction[0]]}')
```

```
Model predicted class is: 4
Model predicted label is: xss
```

Figure 16: Detected as Attack



```
rfc_prediction = rfc_model.predict(df.values)
print(rfc_prediction.tolist())
```

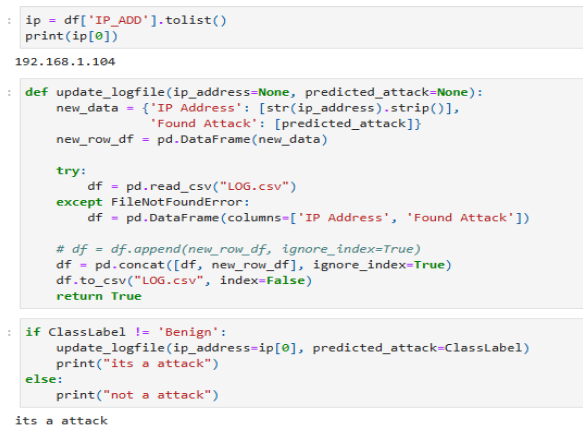
```
[0]
```

```
print(f'Model predicted class is: {rfc_prediction[0]}')
print(f'Model predicted label is: {class_labels[rfc_prediction[0]]}')
```

```
Model predicted class is: 0
Model predicted label is: Benign
```

Figure17: Detected as non Attack

The inference system scans the incoming data, distinguishes between an attack and a non-attack and, if an attack is identified, writes the IP address to `LOG.cvs' It harvested result is send as a alert email to the admin of the threat.



```
ip = df['IP_ADD'].tolist()
print(ip[0])
```

```
192.168.1.104
```

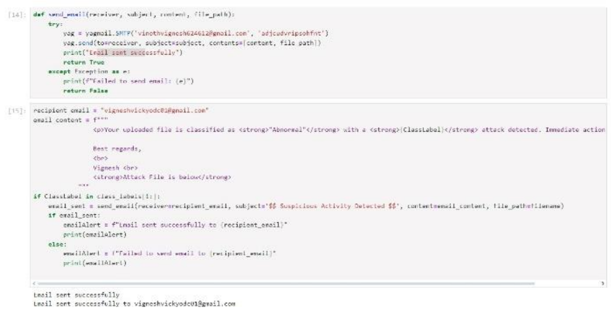
```
def update_logfile(ip_address=None, predicted_attack=None):
    new_data = {'IP Address': [str(ip_address).strip()],
                'Found Attack': [predicted_attack]}
    new_row_df = pd.DataFrame(new_data)

    try:
        df = pd.read_csv("LOG.csv")
    except FileNotFoundError:
        df = pd.DataFrame(columns=['IP Address', 'Found Attack'])

    # df = df.append(new_row_df, ignore_index=True)
    df = pd.concat([df, new_row_df], ignore_index=True)
    df.to_csv("LOG.csv", index=False)
    return True
```

```
if ClassLabel != 'Benign':
    update_logfile(ip_address=ip[0], predicted_attack=ClassLabel)
    print("its a attack")
else:
    print("not a attack")
```

```
its a attack
```

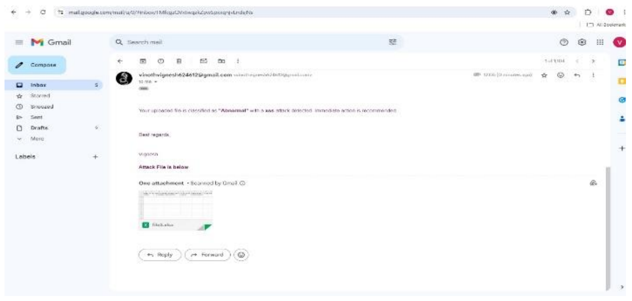
Figure 18: Snips of Logging the result

As for e-mail notifications, the script provides a function send_email to send an e-mail with the help of yagmail. This function tries to send an email and capture any errors or exceptions that may arise. If the predicted as an attack (i. e. , any label other than ‘Benign’), the script sends an email to a specified recipient (vigneshvickyodc01@gmail.com). The email has a subject, body and the original Excel document in the form of an attachment. The below figure 19 & figure 20 illustrates what happens in the case of an attack.



```
[14]: def send_email(receiver, subject, content, file_path):
          try:
              yag = yagmail.SMTP('vinothvignesh624612@gmail.com', 'adjcudvripsohfnt')
              yag.send(to=receiver, subject=subject, contents=[content, file path])
              print('Email sent successfully')
              return True
          except Exception as e:
              print(f"Failed to send email: {e}")
              return False

[15]: recipient email = "vigneshvickyodc01@gmail.com"
      email content = f"""
                  <p>Your uploaded file is classified as <strong>"Abnormal"</strong> with a <strong>{ClassLabel}</strong> attack detected. Immediate action

                  Best regards,
                  <br>
                  Vignesh <br>
                  <strong>Attack File is below</strong>
                """
      if ClassLabel in class_labels[1:]:
          email_sent = send_email(receiver=recipient_email, subject='$$ Suspicious Activity Detected $$', content=email_content, file_path=filename)
          if email_sent:
              emailAlert = f"Email sent successfully to {recipient_email}"
              print(emailAlert)
          else:
              emailAlert = f"Failed to send email to {recipient_email}"
              print(emailAlert)

      Email sent successfully
      Email sent successfully to vigneshvickyodc01@gmail.com
```

Figure 19: Attack happed



vinothvignesh624612@gmail.com

Your uploaded file is classified as "Abnormal" with a xss attack detected. Immediate action is recommended.

Best regards,

Vignesh

Attack File is below

One attachment • Scanned by Gmail

Figure20: Result

6.5 Discussion

While comparing three machine learning models for intrusion detection: SVC, ANN, and RFC, the latter proved to be the most efficient with an average accuracy of 96%. This high accuracy was seen in other network activities such as the identification of benign activities, passwords attacks, scans activity and DoS attacks, which indicates the stability and efficacy of RFC in high volume data sets. Conversely, the ANN model, though possessing the capability to capture intricate relationship hierarchy, obtained only 78% accuracy, indicating that the method and training adopted in this study is lack of optimality. When it came to detecting certain type of attacks such as the DoS and Cross-Site Scripting (XSS) the SVC model hit its limitations and this revealed how sensitive the model is to the intricacies of attack patterns. Making reference to IDS implementation in real-life systems, it is clear that the choice of the model has implications for the detection of threats in an environment.

The better performance of RFC implies its applicability in various primary defense roles in IDS environments particularly where false negatives are expensive. Nonetheless, both insights from the ANN and SVC models highlight that there is a need for further research in refining and improving the models, as well as the utilization of synergistic techniques to amalgamate the best features of the algorithms for optimal detection. Possible directions for the future research include more complex architectures and training algorithms for ANN, the ways of improving the methods of models combination and the enlargement of the dataset with new, more contemporary and diverse attacks for more accurate examination of the model effectiveness against new forms of cyber threats.

7. Conclusion and Future Work

This research aims at comparing three machine learning algorithms, namely, SVC, ANN, and RFC within the context of network intrusion detection. The main research question asked was how these models perform in terms of identifying various forms of cyber threats; this included the non-threatening activities, DoS attacks, XSS, as well as other attacks. The goals were to assess and compare the accuracy, precision, recall, and F1-scores of these models, in an effort to understand their efficiency and to identify their strengths and weaknesses. The study also shows that the RFC model has the highest accuracy level of 96%, suggesting it has high accuracy rates in most of the attack types. The RFC's high precision and recall in identifying benign activities and DoS attacks speak to its applicability to real-world scenarios.

On the other hand, the SVC model had a moderate accuracy of 78.20% where it distinguished effectively the benign activities but it failed in identifying the DoS and XSS attacks. Specifically, the ANN model achieved 78.14% accuracy and showed reasonable results in the detection of benign and scanning activity but struggled with DoS and XSS. Since RFC has been found effective, it can also be used as a strong defense mechanism for IDS, especially when accuracy is a top priority. Nevertheless, the limitations identified in SVC and ANN point to the need for further enhancement and optimization. Specifically, overfitting tendency of the ANN and the variability of the SVC depending on the kind of attack has been identified as areas of concern. The future research should aim at the following;

Improving the ANN model could involve seeking more complex architectures and training methods to reduce overfitting and increase accuracy. Examining how multiple models like SVC and ANN are combined with RFC to form ensembles may provide a workflow that optimizes each. Furthermore, growing the dataset to encompass new and varied attack types will be important in keeping the models pertinent and protective against novel cyber threats. Such developments will help in improving intrusion detection systems and making them more robust and resistant against the existing problems and barriers to the general cybersecurity solutions.

Reference

Abbas, A., Khan, M.A., Latif, S., Ajaz, M., Shah, A.A. and Ahmad, J., 2022. A new ensemble-based intrusion detection system for internet of things. *Arabian Journal for Science and Engineering*, pp.1-15.

Alfarshouti, A.M. and Almutairi, S.M., 2022. An intrusion detection system in IoT environment using KNN and SVC classifiers. *Webology*, *19*(1), pp.3500-3517.

Altulaihan, Esra, Mohammed Amin Almaiah, and Ahmed Aljughaiman. "Anomaly Detection IDS for Detecting DoS Attacks in IoT Networks Based on Machine Learning Algorithms." *Sensors* 24.2 (2024): 713.

Arunkumar, J.R., Velmurugan, S., Chinnaiah, B., Charulatha, G., Prabhu, M.R. and Chakkaravarthy, A.P., 2023. Logistic Regression with Elliptical Curve Cryptography to Establish Secure IoT. *Computer Systems Science & Engineering*, *46*(1).

Asif, Muhammad, et al. "MapReduce based intelligent model for intrusion detection using machine learning

technique." *Journal of King Saud University-Computer and Information Sciences* 34.10 (2022): 9723-9731.
Elrawy, M.F., Awad, A.I. and Hamed, H.F., 2018. Intrusion detection systems for IoT-based smart environments: a survey. *Journal of Cloud Computing*, *7*(1), pp.1-20.
Gad, A.R., Nashat, A.A. and Barkat, T.M., 2021. Intrusion detection system using machine learning for vehicular ad hoc networks based on ToN-IoT dataset. *IEEE Access*, *9*, pp.142206-142217.
Gu, Jie, and Shan Lu. "An effective intrusion detection approach using SVC with naïve Bayes feature embedding." *Computers & Security* 103 (2021): 102158.
Jeevaraj, D., 2023. Feature selection model using naive bayes ML algorithm for WSN intrusion detection system. *International journal of electrical and computer engineering systems*, *14*(2), pp.179-185.
Khan, Muhammad Ashfaq. "HCRNNIDS: Hybrid convolutional recurrent neural network-based network intrusion detection system." *Processes* 9.5 (2021): 834.
Kiran, K.S., Devisetty, R.K., Kalyan, N.P., Mukundini, K. and Karthi, R., 2020. Building a intrusion detection system for IoT environment using machine learning techniques. *Procedia Computer Science*, *171*, pp.2372-2379.
Krzysztoń, Mateusz, and Michał Marks. "Simulation of watchdog placement for cooperative anomaly detection in bluetooth mesh intrusion detection system." Simulation Modelling Practice and Theory 101 (2020): 102041.
Laghrissi, FatimaEzzahra, et al. "Intrusion detection systems using long short-term memory (LSTM)." *Journal of Big Data* 8.1 (2021): 65.
Lin, Zilong, Yong Shi, and Zhi Xue. "Idsgan: Generative adversarial networks for attack generation against intrusion detection." *Pacific-asia conference on knowledge discovery and data mining*. Cham: Springer International Publishing, 2022.
Liu, G., Zhao, H., Fan, F., Liu, G., Xu, Q. and Nazir, S., 2022. An enhanced intrusion detection model based on improved kNN in WSNs. *Sensors*, *22*(4), p.1407.
Logeswari, G., S. Bose, and T. J. I. A. Anitha. "An intrusion detection system for sdn using machine learning." *Intelligent Automation & Soft Computing* 35.1 (2023): 867-880.
Nie, Fengyuan, et al. "M2VT-IDS: A multi-task multi-view learning architecture for designing IoT intrusion detection system." *Internet of Things* 25 (2024): 101102.
Otoum, Yazan, Dandan Liu, and Amiya Nayak. "DL-IDS: a deep learning–based intrusion detection framework for securing IoT." *Transactions on Emerging Telecommunications Technologies* 33.3 (2022): e3803.
Paya, Antonio, et al. "Apollon: a robust defense system against adversarial machine learning attacks in intrusion detection systems." *Computers & Security 136* (2024): 103546.
Sicato, J.C.S., Singh, S.K., Rathore, S. and Park, J.H., 2020. A comprehensive analyses of intrusion detection system for IoT environment. *Journal of Information Processing Systems*, *16*(4), pp.975-990.
Syamsuddin, I. and Barukab, O.M., 2022. SUKRY: suricata IDS with enhanced kNN algorithm on raspberry Pi for classifying IoT botnet attacks. *electronics*, *11*(5), p.737.
Vishwakarma, M. and Kesswani, N., 2023. A new two-phase intrusion detection system with Naïve Bayes machine learning for data classification and elliptic envelop method for anomaly detection. *Decision Analytics Journal*, *7*, p.100233.
Yang, Li, Abdallah Moubayed, and Abdallah Shami. "MTH-IDS: A multitiered hybrid intrusion detection system for internet of vehicles." *IEEE Internet of Things Journal* 9.1 (2021): 616-632.
Yasotha, B., Sasikala, T. and Krishnamurthy, M., 2023. Wrapper Based Linear Discriminant Analysis (LDA) for Intrusion Detection in IIoT. *Comput. Syst. Sci. Eng.*, *45*(2), pp.1625-1640.
Zheng, D., Hong, Z., Wang, N. and Chen, P., 2020. An improved LDA-based ELM classification for intrusion detection algorithm in IoT application. Sensors, 20(6), p.1706