

# Multi-Stage Intrusion Detection System for IoT Network using Deep learning



By

**Dawood Irfan**

**Fall 2019-MS(IT)- 00000319497**

Supervisor

**Dr. Arsalan Ahmed**

**Department of Computer Sciences**

A thesis submitted in partial fulfillment of the requirements for the degree  
of Masters of Science in Information Technology (MS IT)

In

School of Electrical Engineering and Computer Sciences (SEECS)

National University of Sciences and Technology (NUST)

Islamabad, Pakistan

October 2022

## THESIS ACCEPTANCE CERTIFICATE

Certified that final copy of MS/MPhil thesis entitled "Multi-Stage Intrusion Detection System for IoT Network using Deep learning" written by DAWOOD IRFAN, (Registration No 00000319497), of SEecs has been vetted by the undersigned, found complete in all respects as per NUST Statutes/Regulations, is free of plagiarism, errors and mistakes and is accepted as partial fulfillment for award of MS/M Phil degree. It is further certified that necessary amendments as pointed out by GEC members of the scholar have also been incorporated in the said thesis.

Signature: \_\_\_\_\_ 

Name of Advisor: Dr. Arsalan Ahmad

Date: 27-Aug-2022

HoD/Associate Dean: \_\_\_\_\_

Date: \_\_\_\_\_

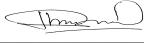
Signature (Dean/Principal): \_\_\_\_\_

Date: \_\_\_\_\_

## Approval

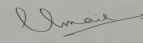
It is certified that the contents and form of the thesis entitled "Multi-Stage Intrusion Detection System for IoT Network using Deep learning" submitted by DAWOOD IRFAN have been found satisfactory for the requirement of the degree

Advisor : Dr. Arsalan Ahmad

Signature:  \_\_\_\_\_

Date: 27-Aug-2022

Committee Member 1: Dr. Umair Hashmi

Signature:  \_\_\_\_\_

Date: 28-Aug-2022

Committee Member 2: Dr. Hassaan Khaliq Qureshi

Signature:  \_\_\_\_\_

Date: 27-Aug-2022

Committee Member 3: Dr. Sajjad Hussain

Signature:  \_\_\_\_\_

Date: 26-Aug-2022

# Dedication

I want to dedicate this research document to my parents, family and teachers who have supported me through every thick and thin.

## Certificate of Originality

I hereby declare that this submission titled "Multi-Stage Intrusion Detection System for IoT Network using Deep learning" is my own work. To the best of my knowledge it contains no materials previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any degree or diploma at NUST SEecs or at any other educational institute, except where due acknowledgement has been made in the thesis. Any contribution made to the research by others, with whom I have worked at NUST SEecs or elsewhere, is explicitly acknowledged in the thesis. I also declare that the intellectual content of this thesis is the product of my own work, except for the assistance from others in the project's design and conception or in style, presentation and linguistics, which has been acknowledged. I also verified the originality of contents through plagiarism software.

Student Name: DAWOOD IRFAN

Student Signature: \_\_\_\_\_



# Acknowledgments

All praises to Almighty Allah for giving me the strength, commitment and persistence to conduct my research work in My Master's degree. I want to express my gratitude to my mentor Dr. Umair Hashmi, who gave me unwavering support in order to bring out the best in me. The results of this research would not have been possible without his assistance, direction, and ongoing encouragement. I sincerely appreciate the assistance of every committee member who helped me during the research process.

**Dawood Irfan**

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Overview . . . . .	1
1.2	Thesis overview . . . . .	4
<b>2</b>	<b>Literature Review</b>	<b>6</b>
<b>3</b>	<b>Methodology</b>	<b>12</b>
3.1	Data Sets . . . . .	12
3.2	Evaluation Metrics . . . . .	15
3.3	Oversampling (SMOTE) . . . . .	16
3.4	DeepInsight . . . . .	17
<b>4</b>	<b>3 Stage Intrusion Detection System</b>	<b>20</b>
4.1	Stage 1 . . . . .	20
4.1.1	Information Gain . . . . .	21
4.2	Stage 2 . . . . .	23
4.2.1	Models for each Dataset . . . . .	23

4.2.2	Model on combined Dataset . . . . .	28
4.2.3	Feedback from the previous stage . . . . .	30
4.3	Stage 3 . . . . .	30
4.3.1	Model on Combined Data . . . . .	30
4.3.2	SMOTE (Oversampling) . . . . .	32
4.3.3	Category DDoS . . . . .	32
4.3.4	Category DoS . . . . .	35
4.3.5	Category Reconnaissance . . . . .	36
4.3.6	Category Theft . . . . .	39
4.3.7	Category Mirari . . . . .	42
4.3.8	Category Scan . . . . .	44
<b>5</b>	<b>Transfer Learning</b>	<b>48</b>
5.1	Base Model . . . . .	50
5.2	Transfer Learning Model . . . . .	51
<b>6</b>	<b>Conclusion and Future Work</b>	<b>53</b>
6.1	Conclusion . . . . .	53
6.2	Future Work . . . . .	53
	<b>References</b>	<b>55</b>



# List of Figures

3.1	The class distribution of BoTnet Data . . . . .	12
3.2	Environment Setup . . . . .	14
3.3	Class distribution of IoT data . . . . .	14
3.4	Image transformation . . . . .	18
3.5	Tabular to Image transformation steps in DeepInsight . . . . .	18
4.1	Classification report of Stage 1 model . . . . .	21
4.2	Confusion Matrix of stage 1 model . . . . .	21
4.3	Information Gain (dataset 1) . . . . .	22
4.4	Information Gain (dataset 2) . . . . .	22
4.5	Classification report of model trained on dataset 2 . . . . .	24
4.6	Classification report of re-tuned model . . . . .	24
4.7	Confusion matrix of retuned model . . . . .	25
4.8	Feature Importance of retuned model . . . . .	25
4.9	Classification report of RFT model . . . . .	26

4.10	Classification report of RFT (Optimized) model	27
4.11	Feature importance of RFT model . . . . .	27
4.12	Classification report of RFT model . . . . .	28
4.13	Confusion matrix . . . . .	29
4.14	Confusion matrix . . . . .	29
4.15	Classification report of model trained on com- bined data . . . . .	31
4.16	Feature importance of model trained on com- bined data . . . . .	31
4.17	Classification report of DDoS model . . . . .	33
4.18	Confusion matrix of DDoS model . . . . .	34
4.19	Feature importance of DDoS model . . . . .	34
4.20	Classification report of DoS model . . . . .	35
4.21	Confusion matrix of DoS model . . . . .	36
4.22	Feature Importance of DoS model . . . . .	36
4.23	Classification report of Reconnaissance model	37
4.24	Confusion matrix of Reconnaissance model . .	38
4.25	Feature Importance of Reconnaissance model .	38
4.26	Classification report of Theft model . . . . .	39
4.27	Confusion matrix of Theft model . . . . .	40
4.28	Feature Impotance of Theft model . . . . .	40

4.29	Classification report of Theft model (top 10 influencing features) . . . . .	41
4.30	Classification report of Mirari model . . . . .	42
4.31	Feature importance of Mirari model . . . . .	43
4.32	Class distribution on most influencing feature . . . . .	43
4.33	Classification report of Scan model . . . . .	45
4.34	Feature importance of Scan model . . . . .	45
5.1	Imgae Data . . . . .	48
5.2	Classification report of model . . . . .	49
5.3	Image Date in 8x8x3 dimmensions . . . . .	50
5.4	Classification report of base model . . . . .	51
5.5	Base model training . . . . .	51
5.6	Classification report of TL model . . . . .	52
5.7	TL model training . . . . .	52

# List of Tables

3.1	Labeled Features . . . . .	13
4.1	Oversampled Classes . . . . .	32
4.2	Stage 3 models performance summary . . . . .	46
5.1	Samples of each class . . . . .	50

# Abstract

With the advent of IoT, an advanced era of communication has been developed as everything around us could be connected to a network. The last decade has seen a growing trend towards developing and deploying a large scale of IoT devices. In general, a typical IoT network will exchange an enormous amount of data every second, thus, these devices are prone to security threats. An intrusion Detection System (IDS) is one of the most common security solutions for identifying cyber-malicious attacks and threats. However, the main challenge faced by many IDSs is the endless increase of new threats that current systems do not recognize.

This project is divided into two parts. In the first part, we aim to introduce multi-stage and efficient intrusion detection system in IoT networks using supervised machine learning (ML). The system proposed will be able to learn the discriminative feature representation automatically from a large amount of data and then accurately classifies the different classes efficiently. In order to achieve this project, we build a model to see whether or not the traffic encountered in the network is normal. Then, the system proposed will detect the different attack classes and sub-classes at different stages. Stage 1

classifies the network packet as normal or anomaly, stage 2 categorizes the category of attack, and stage 3 classifies the sub-category. We plan to validate the robustness and effectiveness of the system proposed using well-known IoT benchmark datasets. After that, our model will be evaluated using different performance metrics and compared with the state-of-the-art techniques to demonstrate its improvements over other related systems.

In the second part, transfer learning is used to improve the performance of the target domain model. To increase the availability of our prior knowledge about the target future data, real-world applications can potentially integrate related data from a different domain in addition to data in the target domain. By transferring valuable information from data in a related domain for use in the target activities, transfer learning solves such cross-domain learning difficulties. The tabular data is converted into images using DeepInsight. Then CNN model is trained on dataset 1. This model is then used as a base model and further trained on sparse dataset 2 in order to determine the knowledge (weights) learned from dataset 1 assist target model to perform better on the target data set.

# Introduction

## 1.1 Overview

The use of IoT devices has significantly increased during the last several years. IoT links the device to the internet so that data can be shared. Through the internet, it may be accessed and operated at any time and from any location. The merging of the internet and physical devices is known as the "internet of things". Intrusion detection is the process of examining network sources and smart device data for unwanted activity. Security is jeopardized when an intrusion occurs in a physical device or network. The system does not reply to the legitimate user as a result of the attacker's illicit actions. By detecting unwanted acts, an intrusion detection system is intended to keep the system secure. IDS outperforms the traditional firewall at detecting intrusions [3]. Host-based or network-based, intrusion detection systems fall under the two categories. The first keeps track of and examines a computer system's internal workings, while the second handles assaults on the communication interfaces

[5]. This project will focus on the former type of intrusion.

The growth of smart devices is the key factor in the interconnected knowledge-based world that is transforming our economies, society, political systems, and critical national infrastructure (CNI). More precisely, IoT devices and smart technologies are crucial to CNI concepts like smart homes, smart cities, intelligent transportation, smart grids, and health care systems. These ideas do, however, come with significant security threats because of their dependence on information and communication technology (ICT) and Internet of Things (IoT) devices, despite the fact that they contribute in everyday tasks [6]. A system may have internal or external intruders. While external intruders do not have access to the system, internal invaders do, however, their rights do not match the level of access they have. These invaders are capable of a wide range of attacks, including user-to-root attacks, denial of service attacks, and probing attacks [5].

The intrusion detection system's capacity to identify new attacks based on previously observed events presents its biggest hurdle. Anomaly-based intrusion detection is ideally suited to the current environment due to the complexity used by attackers and the rise in zero-day assaults [10]. Higher detection rates and fewer false positives are indicators that an IDS is more reliable. An anomaly-based IDS's main goal is to increase the detection rate. A successful cyberattack today has a significant impact on IoT resources, and the time needed to carry one out is getting shorter and shorter. It is anticipated that



very soon, attackers would have the ability to impact IoT networks in a matter of seconds [14].

Deep neural networks (DNNs) have demonstrated notably good performance with audio, text, and images. Canonical designs, which effectively transform raw data into meaningful representations, are what drive this field's quick development. Tabular data is one sort of data that hasn't achieved the same level of success with this type of architecture. Even though tabular data is the most popular format of data (since it includes any category and numerical variables), deep learning for tabular data is largely underdeveloped, and most applications still rely mostly on variations of ensemble decision trees (DTs) [16]. Deep learning for tabular data falls under the following categories i) Data Transformation methods ii) Specialized architectures: DNN architectures tailored toward specific requirements of the heterogeneous tabular data (most approaches of DL for tabular data fall under this category) iii) Regularization models: targets the strong regularization of the learning parameters [17].

The second part of this project uses transfer learning to improve the performance of the model on a sparse dataset. This can be achieved using CNN architectures like VGG16 trained on image data. Instead of using any specialized DNN architecture developed for tabular data i.e. TabNet, the focus is to convert the tabular data into image data using DeepInsight [2], a novel approach to convert tabular data into images. Each row is converted into an RGB image. The converted data is used to train the CNN model.

Due to their exceptional performance in the field of image processing during the past few years, deep learning models, particularly convolutional neural networks, have attained great prominence. By transforming the IoT data into images, the capability of these CNN models may be used to effectively detect complex cyber-attacks [18]. Transfer-learning goal is to improve the performance on target domain when the target data is sparse by transferring the knowledge different but related source domain. Transfer learning has become a well-liked and favorable area in ML because of the numerous application possibilities [20].

## 1.2 Thesis overview

This project comprises of two parts. In the first part three-stage, an intrusion detection system (IDS) is proposed. Stage 1 classifies the network packet as normal or anomalous. If the packet is classified as an anomaly by stage 1 then the flow will be forwarded to stage 2 to classify the category of the attack. Once the category of the flow is classified it will be passed to stage 3 for further sub-category classification. Classification of each label i.e. Label, Category, and Sub-category allows the flexibility to develop different variants of models. For stages 1 and 2, the models are trained on the entire dataset whereas for stage 3 category-specific models are trained as the sub-categories of different categories interact with features differently.

Transfer learning can offer greater advantages to address the target issue with more pertinent data as the big data era progresses [26]. In the second part, the tabular data will be first converted into image format using DeepInsight. Dataset 1 (base data) will be used to train the CNN model. This model will then further train on sparse dataset 2. Prior knowledge (weights) will be used to improve the performance of the target model which is trained on sparse dataset 2.

# Literature Review

A novel distributed IDS system is developed [4] for IoT using deep learning. The modern IDS is proposed [5] name Intrusion Detection CVAE (ID-CVAE) based on a conditional variational autoencoder (CVAE), in which the intrusion labels are incorporated into the CVAE decoder layers. Instead of using the intrusion features as a single input, as is done with a VAE, we employ a generative model based on variational autoencoder (VAE) concepts that relies on two inputs: the intrusion class labels, as well as the intrusion features. Vishwa T et al [7] developed a IDS which is a two-stage detection process, global and local. Local detection is carried out by dedicated sniffers (DS), each of which uses supervised learning techniques based on decision trees to produce instances that are correctly identified. The accumulated measure of fluctuation (AMoF) is a time-based profile that is created by the global stage using the CCIs that are gathered and applied iterative linear regression to the super node (SN) and malicious nodes, respectively. A profile of a malicious and a normal node is obtained, and an anomaly is detected

after three iterations.

Rohan et al [8] developed ID process that demonstrates high accuracy DDoS detection in IoT network traffic can be achieved using a variety of machine learning algorithms, including neural networks, when IoT-specific network behaviors (such as a limited number of endpoints and regular time intervals between packets) are used to guide feature selection. The paper shows that low-cost machine learning algorithms and flow-based traffic data can be used by home gateway routers or other middleboxes to automatically identify local IoT device sources of DDoS attacks. Nevertheless, this experiment focuses only on one attack category i.e. DDoS. A novel IDS system [9] is proposed which is a combination of unsupervised and supervised machine learning models such as K-means and decision tree. The performance of the hybrid model is evaluated against the individual models which come out to be lower. Despite having a lower detection rate than the other two individual techniques, the hybrid IDS is more accurate. The other two IDS experience a larger rate of false positives than the hybrid method, which greatly reduces them. Lowri et al [6] proposed a three-layer IDS system that employs a supervised approach to identify a variety of common network-based cyberattacks against IoT networks. The system has three primary functions and the performance evaluation metric is the F1 score: classifying the type and profile of each IoT device connected to the network with F1 score of 96.2 percent; identifying malicious packets on the network while an attack is occurring with F1 score 90 per-

cent; and classifying the nature of the deployed attack with score 98percent. Imtiaz et al [10] proposed a two-stage intrusion detection system that classifies the network packet as normal or anomaly at stage, and if the packet is classified as an anomaly it will be passed to stage 2 to classify the category and sub-category. Various machine learning models were trained on flow-based features of network data which were extracted from the Botnet dataset with decision tree-based models giving the highest evaluation score.

Ullah et al [11] proposed two-level hybrid model for detecting anomalous packets. The level-1 model employs flow-based anomaly detection, which can distinguish between normal and anomaly network traffic. The CICIDS2017 and UNSW-15 datasets are used to extract the flow-based features. If an anomalous activity is found, the flow is passed on to the level-2 model, which carefully examines the packet's contents to determine the type of anomaly. The level-2 model employs Edited Nearest Neighbors (ENN) to clean the CICIDS2017 and UNSW-15 datasets, SMOTE for oversampling, and Recursive Feature Elimination (RFE) to identify relevant features. Alalade et al [12] introduces Extreme Learning Machine and Artificial Immune System that were used to detect anomalies in the smart home network as part of early work on an IDS (AIS-ELM). Pamukov et al [13] proposed an algorithm for improving the detection misclassification rate. Without operator input, the algorithm can conclusively classify a significant portion of potential invasions as true or false. This algorithm is based on immunology's costimulation theory and

the Negative Selection Algorithm. It implements a co-stimulation strategy intended to reduce the amount of detection errors without requiring operator input via a two-tiered negative selection method. Ullah et al [14] proposed an IoT environment to create realistic benchmark data. The performance of the models is improved by using Recursive feature elimination which selects important features and a correlation coefficient of 0.70 to remove features that are highly correlated. Decision tree-based model achieved the highest evaluation score across all the label features. Ullah et al [15] proposed another IoT dataset extracted from pcap files obtain from a testbed environment that imitates a realistic network environment. Various models are trained after feature selection to show the effectiveness of the proposed dataset.

Sunnyvale [16] proposed TabNet, a deep learning architecture for tabular data. TabNet takes unprocessed raw tabular data as input and use gradient descent to optimize the performance, using the sequential approach at each step in order to decide which feature will impact each decision step. The feature selection is instance-wise. DeepInsight [2] convert non-image (tabular) data into image format which enables the utilization of CNN including GPUs for non-image data.

Faisal et al [18] proposed a data conversion method. Features are normalized (after dropping null and bad values). Then 180 samples are selected iteratively and converted into 60x60x3 RGB images. The ‘normal’ and ‘anomaly’ samples are converted separately and labels

are labeled accordingly. The images are then fed into ResNet architecture for classification. Sun et al [19] propose a similar approach but without feature normalization and swapping.

Yilmaz et al [21] used transfer learning in two settings i) for creating reliable algorithms for new devices via transfer learning and ii) detecting new variants of attack by knowledge transfer. The paper shows improved performance and reduction in training time of the transfer learning approach over the traditional machine learning approach. Chiba et al [22] proposed a transfer learning approach for autonomous driving and shows the effectiveness of this approach. Nalini et al [23] perform case study, various open source pre-trained deep neural networks like GoogleNet, AlexNet and VGG16 are used for transfer learning. The authors show that the models achieve high performance on the input datasets with lower training time.

With the use of deep reinforcement learning, Lui et al [24] proposed an extremely effective IoT network dynamic clustering method for edge computing. The proposed method can satisfy both the load-balancing needs of edge servers and the data transfer needs of IoT networks and is implemented using Deep Q Learning Network. Nageisetty et al [25] presented deep learning framework for detecting complex cyber attacks on an IoT network using Keras. The models used are MLP, CNN, DNN and Autoencoder. Models performance evaluation is performed using well-known IoT benchmark datasets. Phan et al [27] proposed Q-Transfer which maximizes the positive transfer while keeping negative transfer low. The authors show a case



study with a DDoS attack using MLP to showcase the effectiveness of the proposed method.

# Methodology

## 3.1 Data Sets

Datasets 1 is obtained from [14]. Five IoT devices that were run locally and connected to the cloud infrastructure using the node-red tool to produce realistic network traffic were used to model a typical smart home setting. The proposed dataset's attack taxonomy is depicted in Fig. 3.1.

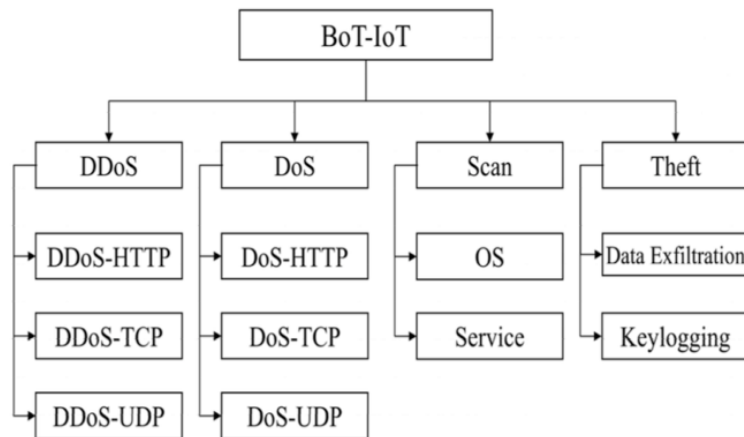


Figure 3.1: The class distribution of BoTnet Data

80 statistical network features are extracted from Pcap files by the ISCXFlow metre. The proposed IoT botnet dataset has three label features in addition to 83 network features. Binary, category, and subcategory are the label features. Table 3.1 contains information regarding each label features. Eight correlated features were separated using 0.70 as the correlation coefficient to exclude the correlated features.

**Table 3.1:** Labeled Features

<b>Binary</b>	<b>Category</b>	<b>Sub-Category</b>
<b>Normal</b>	Normal	DDoS (HTTP, TCP, UDP)
<b>Anomaly</b>	DDoS, Dos, Scan, Theft	DDoS (HTTP, TCP, UDP), OS, Service, Data Exfiltration, Keylogging

Dataset 2 is obtained from [15]. IoT devices and connecting infrastructure make up the testbed for the IoTID20 dataset. The smart home device and the Wi-Fi camera were used to create a typical smart home setting, which produced the IoTID20 dataset. The testbed environment for the proposed dataset is shown in Fig 3.2, whereas, the dataset’s respective attack taxonomy can be seen in Fig. 3.3.

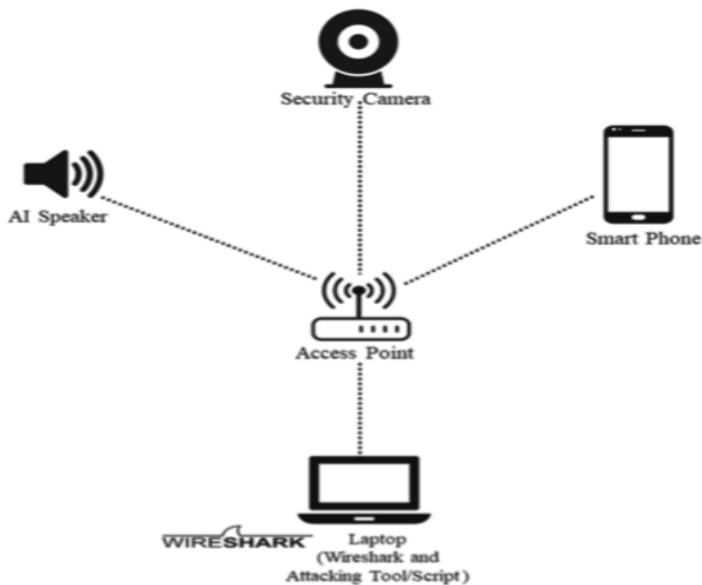


Figure 3.2: Enviroment Setup

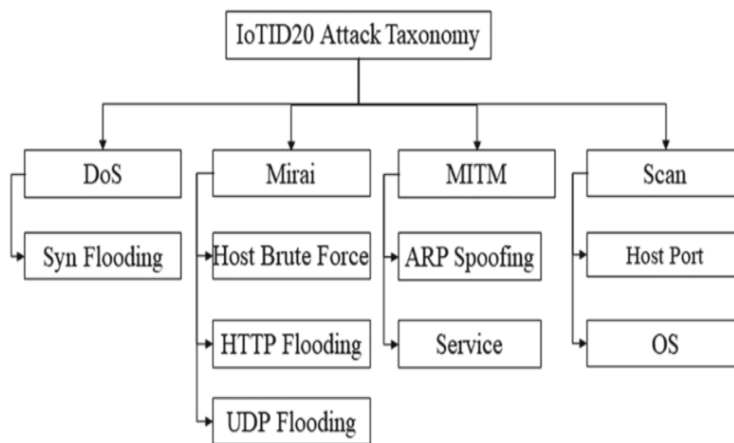


Figure 3.3: Class distribution of IoT data

The authors used CICflowmeter application to extracted the flow-based features from the pcap files. The proposed dataset comprises of three label features and 83 network features. The IoTID20 dataset’s most significant advantages are that it imitate a modern IoT communication trend. A correlation coefficient of 0.70 is used to filter out highly corelated features.

### 3.2 Evaluation Metrics

**Accuracy:** This measures how frequently the classifier makes a correct prediction. The accuracy is defined as the ratio of correct predictions and the sum of predictions.

$$Accuracy = \frac{(TP + TN)}{(TP + FP + TN + FN)}$$

**True Positive (TP):** Predicted positive, actual positive.

**True Negative (TN):** Predicted negative, actual negative.

**False Positive (FP):** Predicted positive, actual negative.

**False Negative (FN):** Predicted negative, actual positive.

**Precision:** This measures the number of actual positives among the predicted positives i.e. how many predicted positives are actually true

$$Precision = \frac{TP}{(TP + FP)}$$

**Recall:** This measures the number of actual positives classifier able to predict.

$$Recall = \frac{TP}{(TP + FN)}$$

**F1-Score:** This balances the tradeoff between precision and recall.

$$F1 - Score = \frac{2 * (Precision * Recall)}{(Precision + Recall)}$$

The target metric in this project is F1-score. Precision means that model will only predict as an anomaly if the probability is higher i.e. to improve the precision the threshold is increased. This will increase the number of FN. Recall measures the ability of the model to classify the maximum number of true as positive i.e. to improve the recall the threshold is decreased. The model will predict the instance as positive even when the probability is low. This will result in an increased number of FP. Both of these metrics are not ideal to evaluate the models when the target is to maximize TP while keeping FN low. To achieve this tradeoff F1-score is used as key classification metric to evaluate the performance of the models.

### **3.3 Oversampling (SMOTE)**

The SMOTE algorithm's main goal is to balance out the data by creating new minority samples. In particular, the SMOTE algorithm performs linear interpolation in minority class samples that are close together. The conventional SMOTE algorithm is briefly described in the paragraphs that follow [1].

For each sample in the minority class, the SMOTE algorithm looks for samples that are close neighbours. And then used linear interpolation to select samples at random from these closest neighbouring samples. Therefore, new samples will be created for each of the original minority samples. Interpolating between the initial minority class samples and its nearby samples is the next stage [1]. The

following is the formula for linear interpolation:

$$X_{new} = X_{origin} + rand(0, 1) * (X_i - X_{origin}), i = 1, 2, \dots, N$$

In the formula above,  $X_{new}$  stands for the newly synthesised minority class sample, whereas  $X_{origin}$  stands for the original samples that were used to generate the new samples.  $X_i$  is a sample that is randomly chosen from the  $k$  neighbouring samples of the minority class sample  $X_{origin}$ , and  $rand(0,1)$  is a random number between 0 and 1 [1].

### 3.4 DeepInsight

The idea behind DeepInsight is to first convert a non-image sample into an image form before feeding it into the deep neural network for classification. Fig 3.4 provides a clear example of how a feature vector  $x$  made up of gene expression values can be translated into a feature matrix  $M$ . The similarity of the features determines where they are located in the Cartesian coordinate system. For instance characteristics  $g_1$ ,  $g_3$ ,  $g_6$ , and  $g_d$  are located closer to one another. The expression values or feature values are mapped after the locations of each feature in a feature matrix have been established. Each sample will provide a different image as a result [2].

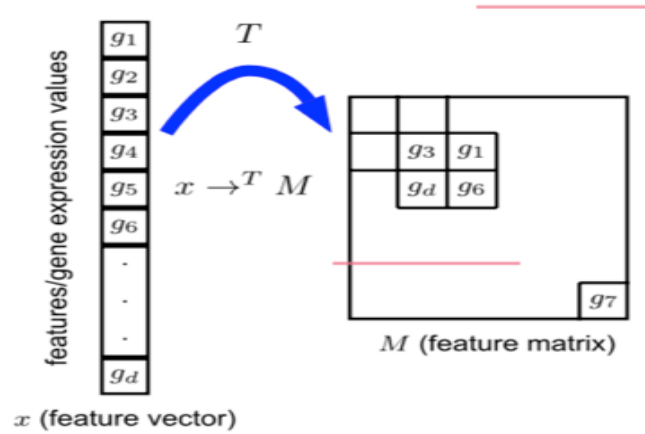


Figure 3.4: Image transformation

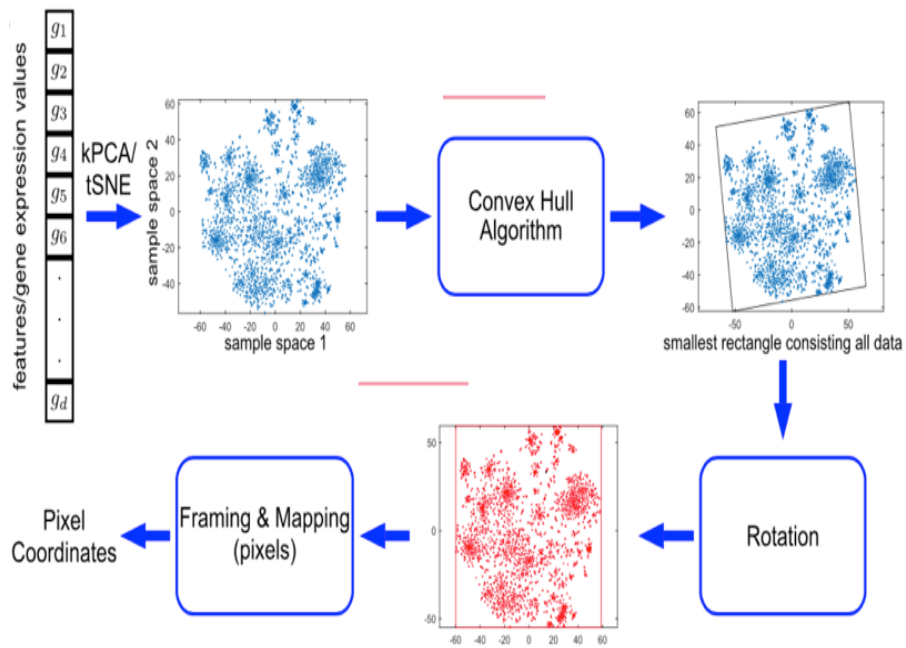


Figure 3.5: Tabular to Image transformation steps in DeepInsight

$Z$  samples of  $m \times n$  feature matrices will be produced from  $Z$  samples of  $d$  features. All of the  $d$  characteristics will be present in this 2D matrix form. The CNN architecture is then used to process this collection of  $N$  feature matrices in order to train the model and provide predictions. The image's single layer includes  $n$  normalized



shades that fall between  $[0, 1]$ . As a result, before executing the image transformation, feature values must be normalised. Two type of normalization can be performed. (1) each feature was assumed to be independent, so its minimum and maximum values were used to normalise it; and (2) the topology of mutual features was somewhat preserved by normalizing it with the maximum value obtained from the entire training set [2].

## 3 Stage Intrusion Detection System

### 4.1 Stage 1

In this stage both datasets are concatenated, and each label class is assigned with different label. For example, for dataset 1 and dataset 2 normal the label will be ‘normal\_1’ and ‘normal\_2’ respectively. Stage 1 not only classify normal and anomaly but also determine the system type. Performance of the model:

$$Testerror = 0.013$$

$$Accuracy = 0.99$$

Random forest tree is used for this stage. The classification report and the confusion matrix are shown in Fig. 4.1 and Fig. 4.2.

F1 score is used as key metric indicator for evaluating the performance of the models. Label ‘Attack\_1’ have the highest F1 score and it also the label with the greatest number of instances in the dataset.

	precision	recall	f1-score	support
Normal_1	1.00	0.85	0.92	19320
Attack_1	0.99	1.00	1.00	368770
Normal_2	1.00	0.59	0.74	8041
Attack_2	0.97	0.99	0.98	117227
accuracy			0.99	513358
macro avg	0.99	0.86	0.91	513358
weighted avg	0.99	0.99	0.99	513358

Figure 4.1: Classification report of Stage 1 model

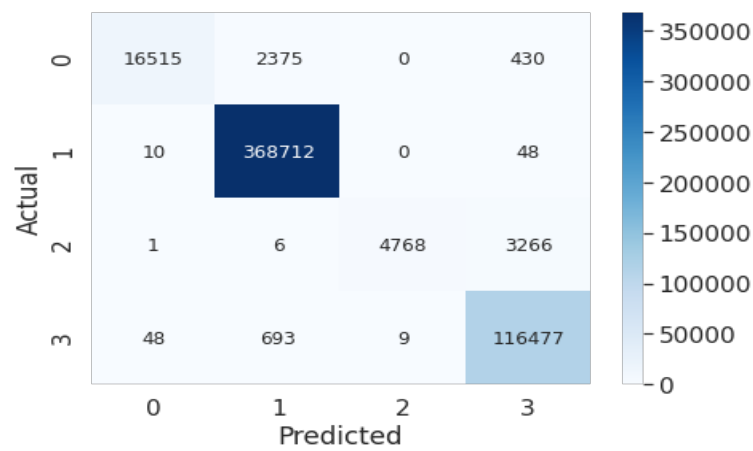


Figure 4.2: Confusion Matrix of stage 1 model

#### 4.1.1 Information Gain

Entropy of target variable ‘Label’:

$$Dataset1 : 0.28$$

$$Dataset2 : 0.34$$

Information gain of each feature is calculated i.e.  $E(\text{Label Feature}(n))$  in order to get intuition which features have greater impact on reducing the impurity of target variables. This indicates the features having more impact on model decision-making.

For dataset 1 the 3 most influential features are ‘Src\_Port’, ‘Bwd\_Pkts/s’ shown in Fig. 4.3 and ‘Flow\_Pkts/s’ and for dataset 2 ‘Dst\_Port’, ‘Src\_Port’, ‘Flow\_Byts/s’ shown in Fig. 4.4. The IG does not necessarily indicate the feature influence on the model but rather give insight to the degree of impact of each feature on target variable.

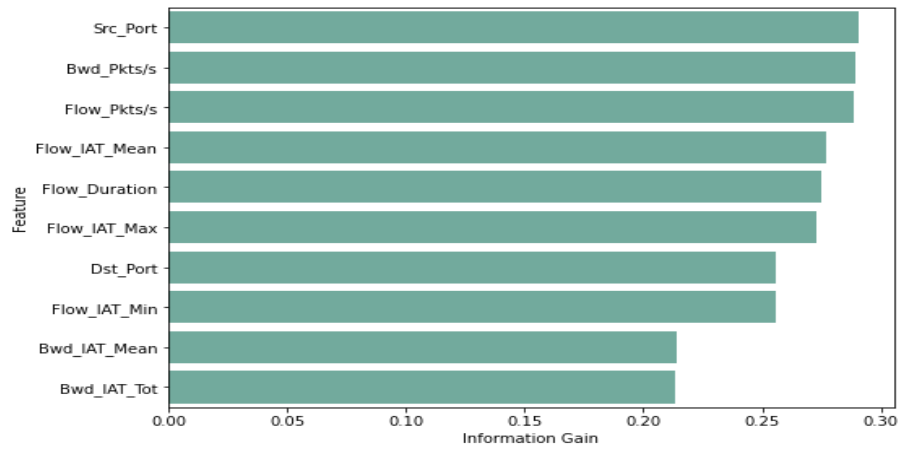


Figure 4.3: Information Gain (dataset 1)

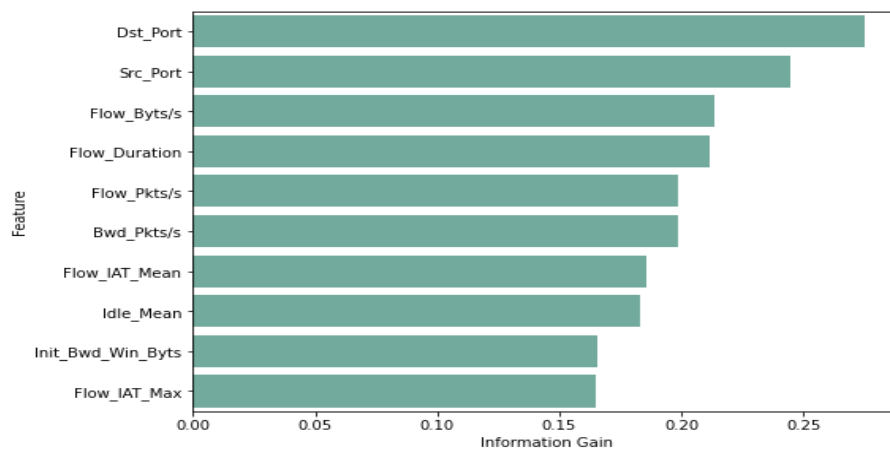


Figure 4.4: Information Gain (dataset 2)

## 4.2 Stage 2

### 4.2.1 Models for each Dataset

#### Model 1 (System 1)

In this case, a model is only trained on the categories of dataset 1. Dataset 1 has four categories:

- DD0s
- Dos
- Reconnaissance
- Thefy

The model is trained using all the features. Following stage 1, Random Forest Tree is used as it gives the best result compared to other models. The performance of RFT is further improved by hyperparameter tuning using grid search.

RFT:

- Accuracy: 0.94

The classification report breakdown of each class is shown in Fig. 4.5. It can be seen from the classification report that the model is performing under par for the category ‘Theft’ with f1 score 0.39 as the number of samples for ‘Theft’ category are considerably less compare to other categories. Next the model is retuned by performing grid search on hyperparameters like ‘number of trees’ and ‘max depth’ to better fit the training data.

Classification Report				
	precision	recall	f1-score	support
DDoS	0.94	0.89	0.92	127267
DoS	0.90	0.95	0.92	130469
Reconnaissance	1.00	0.99	1.00	111434
Theft	1.00	0.24	0.39	113
accuracy			0.94	369283
macro avg	0.96	0.77	0.81	369283
weighted avg	0.94	0.94	0.94	369283

Figure 4.5: Classification report of model trained on dataset 2

RFT (Tuned):

- Accuracy: 0.98

The classification report along with the confusion metric are shown in Fig. 4.6 and Fig. 4.7. With hyperparameter tuning not only is the model accuracy improved the f1 score for each category is also improved. The f1 score for category ‘Theft’ improved from 0.39 to 0.97.

Classification Report				
	precision	recall	f1-score	support
DDoS	0.98	0.95	0.96	127267
DoS	0.95	0.99	0.97	130469
Reconnaissance	1.00	1.00	1.00	111434
Theft	1.00	0.95	0.97	113
accuracy			0.98	369283
macro avg	0.98	0.97	0.98	369283
weighted avg	0.98	0.98	0.98	369283

Figure 4.6: Classification report of re-tuned model

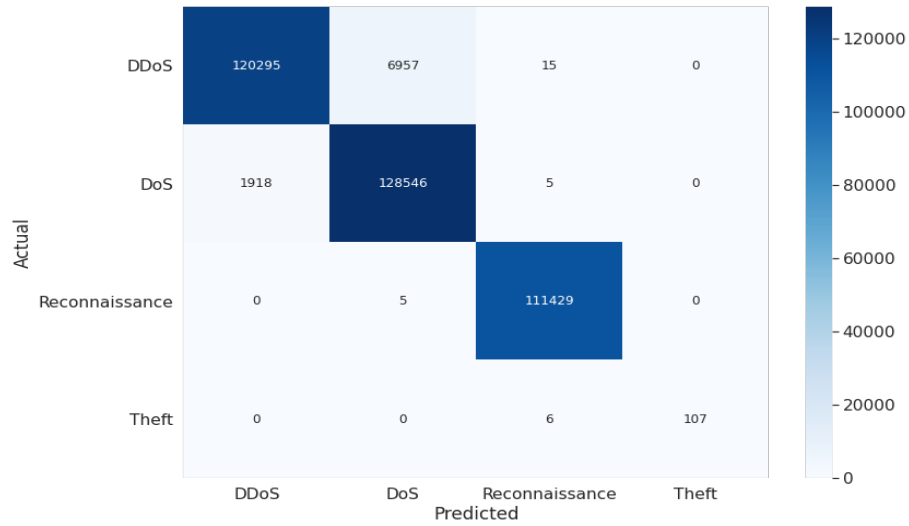


Figure 4.7: Confusion matrix of retuned model

**Feature Importance:** The feature importance of the model is shown in Fig. 4.8. The top 3 influencing features are ‘Flow-Duration’, ‘Bwd\_Pkts/s’ and ‘Flow\_IAT-Max’.

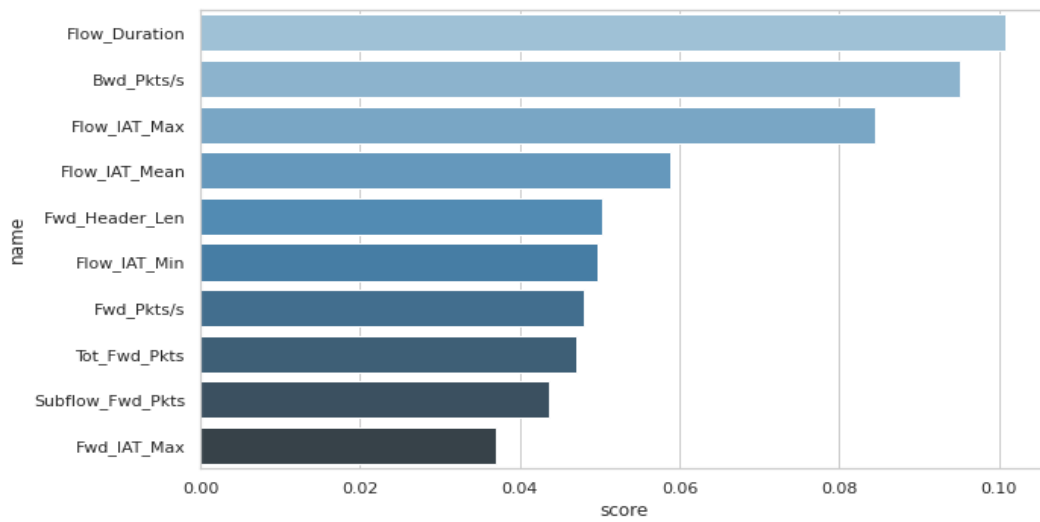


Figure 4.8: Feature Importance of retuned model

**Model 2 (System 2)**

Similarly, this model is only trained on dataset 2. Dataset 2 have 4 categories:

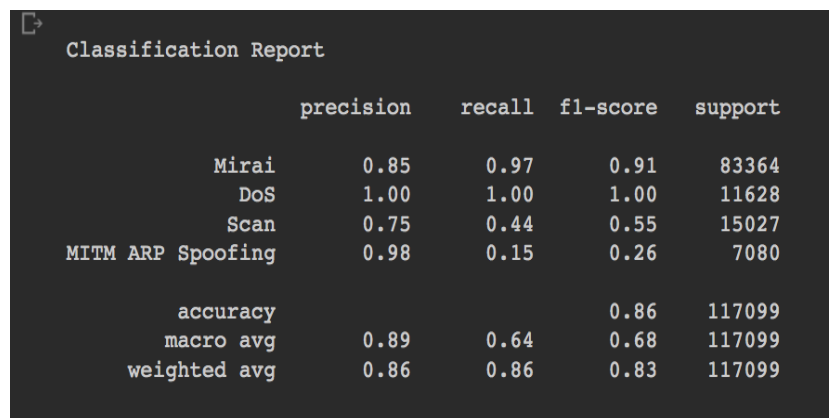
- Dos
- Mitm APR Spoofing
- Mirari
- Scan

Random Forest Tree (RFT) classifier is trained and the performance of the model is further improved via hyperparameter tuning. The performance of the model is:

RFT:

- Accuracy: 0.86

The model performed poor on ‘Mitm APR Spoofing’ with F1 score: 0.26 (Fig. 4.9). This category have the least number of samples in the dataset due to which model struggles too.



```

Classification Report

```

	precision	recall	f1-score	support
Mirai	0.85	0.97	0.91	83364
DoS	1.00	1.00	1.00	11628
Scan	0.75	0.44	0.55	15027
MITM ARP Spoofing	0.98	0.15	0.26	7080
accuracy			0.86	117099
macro avg	0.89	0.64	0.68	117099
weighted avg	0.86	0.86	0.83	117099

**Figure 4.9: Classification report of RFT model**



RFT (tune): The model is tuned by performing hyperparameter tuning via grid search to improve the performance of the model, which can be observed in classification report in Fig. 10.

- Accuracy: 0.92

Classification Report				
	precision	recall	f1-score	support
Mirai	0.92	0.96	0.94	83065
DoS	1.00	1.00	1.00	11810
Scan	0.82	0.89	0.86	14774
MITM ARP Spoofing	0.85	0.28	0.42	7048
accuracy			0.92	116697
macro avg	0.90	0.78	0.80	116697
weighted avg	0.91	0.92	0.91	116697

Figure 4.10: Classification report of RFT (Optimized) model

Even with hyperparameter tuning, RFT is underfitting on the category ‘Mitm Arp Spoofing’. The feature importance of the model is shown in Fig. 4.11. The top three influencing features are ‘Bwd\_IAT\_Min’, ‘SYN\_Flag\_CNT’ and ‘Src\_Port’. The model still underfitting on category ‘Mitm ARP Spoofing’

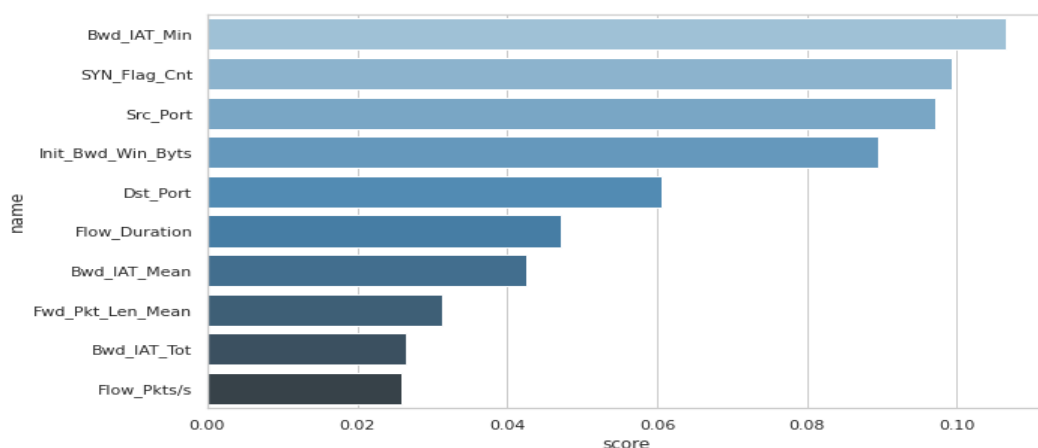


Figure 4.11: Feature importance of RFT model

### 4.2.2 Model on combined Dataset

Both datasets are combined and a label is assigned to each category via label encoding. Feature space is same but the categories are different excluding one. Category ‘Dos’ exist in both datasets but different labels are assigned to them. The accuracy of RFT after hyperparameter tuning is 0.96. The classification report of mentioned model on combined dataset is shown in Fig. 4.12.

	precision	recall	f1-score	support
DDoS	0.98	0.94	0.96	126893
DoS	0.94	0.98	0.96	129759
Reconnaissance	1.00	1.00	1.00	111011
Theft	1.00	0.76	0.87	123
Normal1	1.00	0.98	0.99	19466
Mirai	0.91	0.98	0.94	83606
DoS	1.00	1.00	1.00	11794
Scan	0.88	0.87	0.87	14960
MITM ARP Spoofing	0.90	0.21	0.34	7019
Normal2	0.99	0.81	0.89	8058
accuracy			0.96	512689
macro avg	0.96	0.85	0.88	512689
weighted avg	0.96	0.96	0.96	512689

Figure 4.12: Classification report of RFT model

Whereas, its respective confusion matrix is shown in Fig. 4.13.

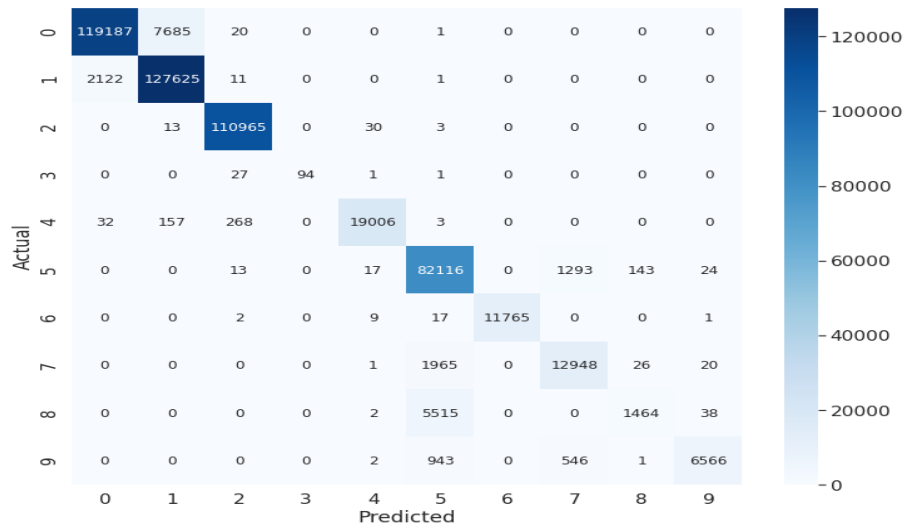


Figure 4.13: Confusion matrix

The performance of the system 1 model is better than the model trained on combined data, but the performance of the model train of system 2 degrades by 4%. The model is still underfitting category ‘Mitm Arp Spoofing’. The lists of the most influencing features for the model trained on dataset 2 and the model trained on the combined dataset are quite similar which explains why the model still underfit the target category.

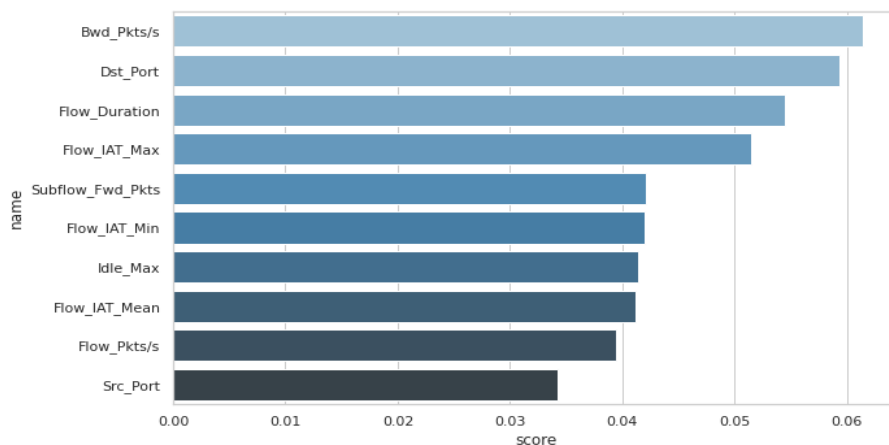


Figure 4.14: Confusion matrix

The top 10 influencing features are shown in Fig. 4.14. Even the most influencing features are weakly correlated with the target variable. Due to this linear models like logistic regression perform poorly on this dataset. To counter this issue decision tree i.e. random forest tree is used to take advantage of the ensemble method and to further optimize the performance via hyperparameters tuning.

### 4.2.3 Feedback from the previous stage

In this process, the feedback from the previous stage is used as a feature to train the model. Stage 1 not only classifies the ‘Normal’ or ‘Anomaly’ but also predicts the system the data point belongs to. This is used as a feature to train this model which indicates which system i.e. 1 or 2 this anomaly belongs to further classify the category. The accuracy of RFT after hyperparameter tuning is 0.96. No improvement in the performance of the model by using feedback from the previous stage

## 4.3 Stage 3

### 4.3.1 Model on Combined Data

Following stage 2, the model is trained on the combined dataset to further classify the sub-category of the anomaly. The number of categories is 19. The accuracy of RFT (tunned) is 0.87. The classification report and feature importance are shown in Fig. 4.15 and Fig. 4.16.

	precision	recall	f1-score	support
DDoS_UDP	0.97	1.00	0.98	59613
DoS_UDP	0.99	0.97	0.98	59773
DDoS_TCP	1.00	0.98	0.99	59544
DoS_TCP	0.98	1.00	0.99	60103
Service	0.97	0.86	0.91	57912
OS	0.86	0.97	0.91	52910
Mirai-UDP Flooding	0.80	0.79	0.79	36710
Normal	0.97	0.93	0.95	27393
Mirai-Hostbruteforceg	0.47	0.94	0.63	24464
DoS-Synflooding	1.00	1.00	1.00	11762
Mirai-HTTP Flooding	0.31	0.33	0.32	11043
Mirai-Ackflooding	0.30	0.22	0.26	10946
Scan Port OS	1.00	0.00	0.00	10585
DoS_HTTP	0.63	0.97	0.76	10222
DDoS_HTTP	0.85	0.20	0.32	7250
MITM ARP Spoofing	0.93	0.16	0.27	6987
Scan Hostport	0.88	0.04	0.08	4457
Keylogging	0.75	0.72	0.74	64
Data_Exfiltration	1.00	0.12	0.21	26
accuracy			0.87	511764
macro avg	0.82	0.64	0.64	511764
weighted avg	0.89	0.87	0.86	511764

Figure 4.15: Classification report of model trained on combined data

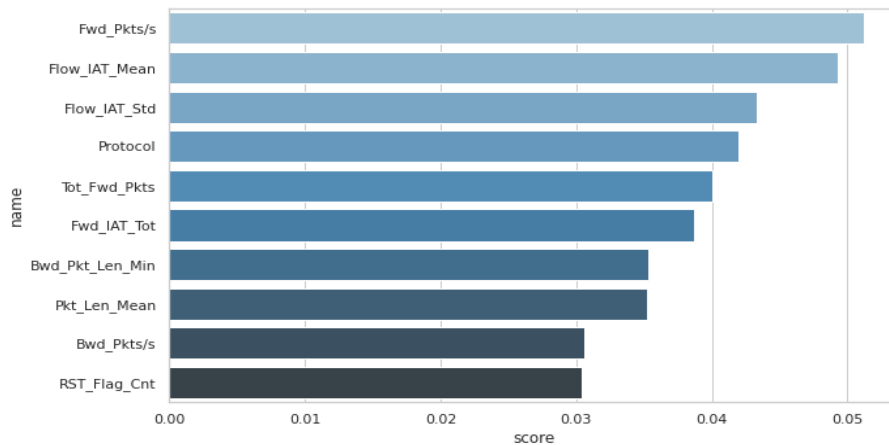


Figure 4.16: Feature importance of model trained on combined data

The accuracy of the model is above par and this is due to the fact that model performs well on majority classes but underfit the minority classes. As the model shows high performance on the majority classes, the model's overall accuracy improved. One way to resolve this is to use an oversampling technique like SMOTE to add more data points of the minority class so the model have more data to learn meaningful relation between the features and the minority class.

### 4.3.2 SMOTE (Oversampling)

Four sub-categories are oversampled using smote. The detail of oversampling is shown in Table 4.1.

**Table 4.1:** Oversampled Classes

Classes	Actual Instances	Over sampled Instances
<b>DoS_HTTP</b>	51,112	150,000
<b>DDoS_HTTP</b>	36,539	100,000
<b>Keylogging</b>	384	50,000
<b>Data_Exfiltration</b>	136	50,000

The accuracy of RFT is 0.81, down from 0.87 on original combine dataset. This indicates that simply adding more examples of minority class doesn't help the model to extract meaningful relation among features and minority classes. With this many classes, the features are very weakly correlated with the target variable i.e. the most influencing feature 'Fwd\_Pkts/s' score is 0.06 as shown in fig 4.16. The model is unable to map the relationship between the features and the classes. To counter this, category-specific models are trained. These models are only trained to classify the subcategories of a specific category.

### 4.3.3 Category DDoS

The subcategories are:

- DDoS\_UDP
- DDoS\_TCP
- DDoS\_HTTP

The performance of the models are:

- Logistic Regression
  - Accuracy: 0.94
- RFT (tuned)
  - Accuracy: 1.00

The classification report and confusion matrix of the RFT (tuned) model are shown in Fig. 4.17 and Fig. 4.18. Confusion metric shows no error in prediction by the model on test data and hence the F1 score for all the subcategories is 1.

Classification Report				
	precision	recall	f1-score	support
DDoS_UDP	1.00	1.00	1.00	60080
DDoS_TCP	1.00	1.00	1.00	59869
DDoS_HTTP	1.00	1.00	1.00	7305
accuracy			1.00	127254
macro avg	1.00	1.00	1.00	127254
weighted avg	1.00	1.00	1.00	127254

Figure 4.17: Classification report of DDoS model

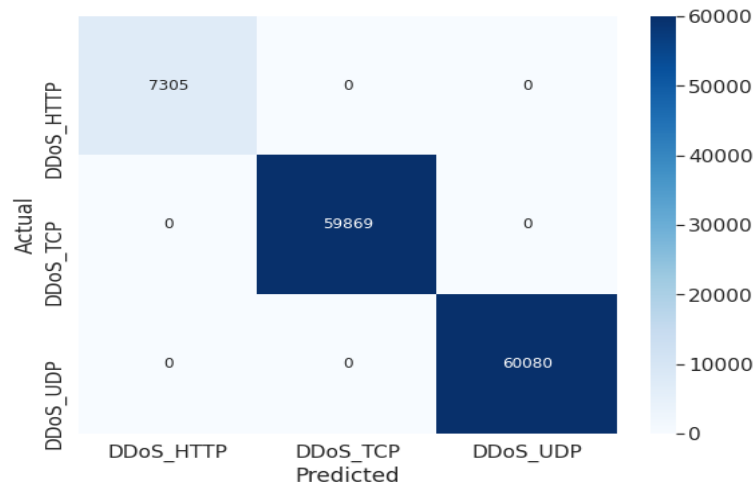


Figure 4.18: Confusion matrix of DDoS model

The feature’s importance is shown in Fig. 4.19. The most influencing feature i.e. ‘Pkt\_Len\_Mean’ has a high score of 0.14. By training the model on a specific category only assist the model in learning meaningful relation between features and the target variable.

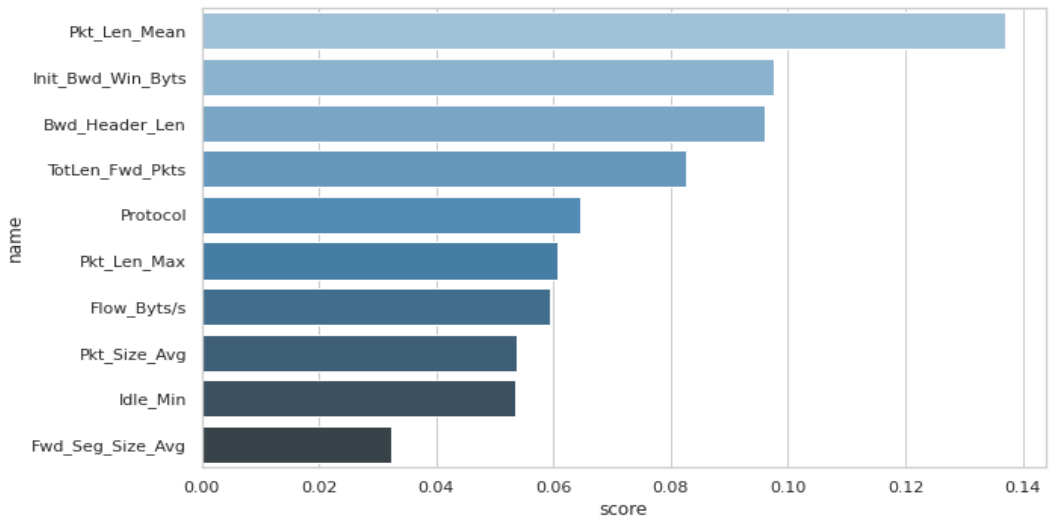


Figure 4.19: Feature importance of DDoS model



#### 4.3.4 Category DoS

The sub-categories are:

- DoS\_UDP
- DoS\_TCP
- DoS\_HTTP

The performance of the models are:

- Logistic Regression
  - Accuracy: 0.92
  - TPR: 0.92
  - FPR: 0.06
- RFT (tuned)
  - Accuracy: 1.00

The classification report along with the confusion matrix are shown in Fig. 4.20 and Fig. 4.21. This model also achieves the perfect score of 1 as all of the data points of test data are classified correctly.

Classification Report				
	precision	recall	f1-score	support
DoS_TCP	1.00	1.00	1.00	60082
DoS_UDP	1.00	1.00	1.00	60374
DoS_HTTP	1.00	1.00	1.00	10120
accuracy			1.00	130576
macro avg	1.00	1.00	1.00	130576
weighted avg	1.00	1.00	1.00	130576

Figure 4.20: Classification report of DoS model

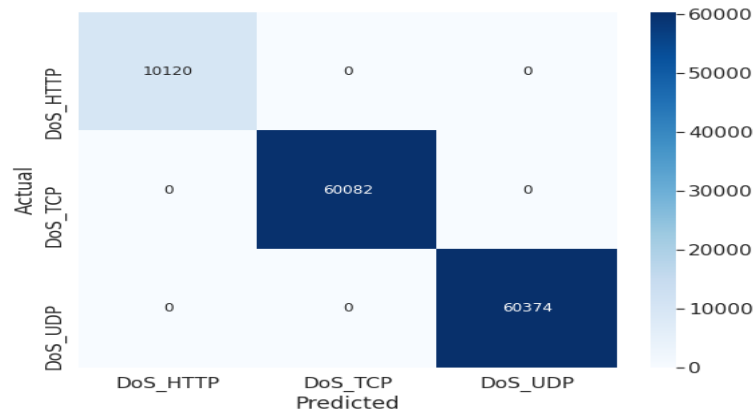


Figure 4.21: Confusion matrix of DoS model

The list of the 10 most influencing features for this model is different (Fig. 4.22). The most influencing feature is ‘Pkt\_Len\_Max’ with score of 0.12 followed by ‘Pkt\_Len\_Mean’ and ‘Init\_Bwd\_Win\_Byts’ with scores of 0.11 and 0.08 respectively.

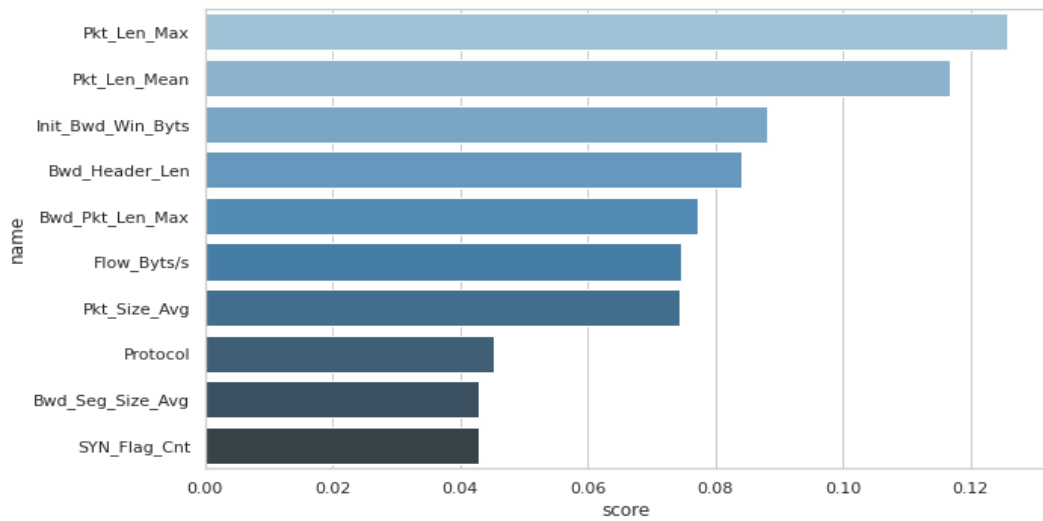


Figure 4.22: Feature Importance of DoS model

### 4.3.5 Category Reconnaissance

- OS
- Service

The performance of the models are:

- Logistic Regression
  - Accuracy: 0.52
  - TPR: 0.52
  - FPR: 0.52
- SVM
  - Accuracy: 0.86
- RFT (tuned)
  - Accuracy: 0.93

The model achieves higher precision for sub-category ‘Service’ while for ‘OS’ higher score for recall. The classification report and confusion matrix are shown in Fig. 4.23 and Fig. 4.24

Classification Report				
	precision	recall	f1-score	support
Service	0.98	0.88	0.93	58471
OS	0.89	0.98	0.93	53014
accuracy			0.93	111485
macro avg	0.93	0.93	0.93	111485
weighted avg	0.94	0.93	0.93	111485

Figure 4.23: Classification report of Reconnaissance model

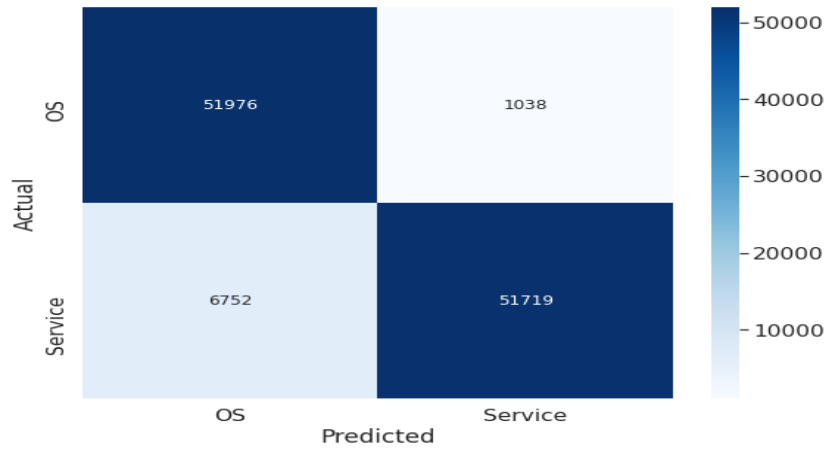


Figure 4.24: Confusion matrix of Reconnaissance model

Again the list of most influencing features is different as compared to other models of trained on different categories of same dataset. The most influencing feature for the model is 'Bwd\_IAT\_Mean' with score of 0.13 as shown in Fig. 4.25. The rest of the influencing features can be seen in Fig. 4.25.

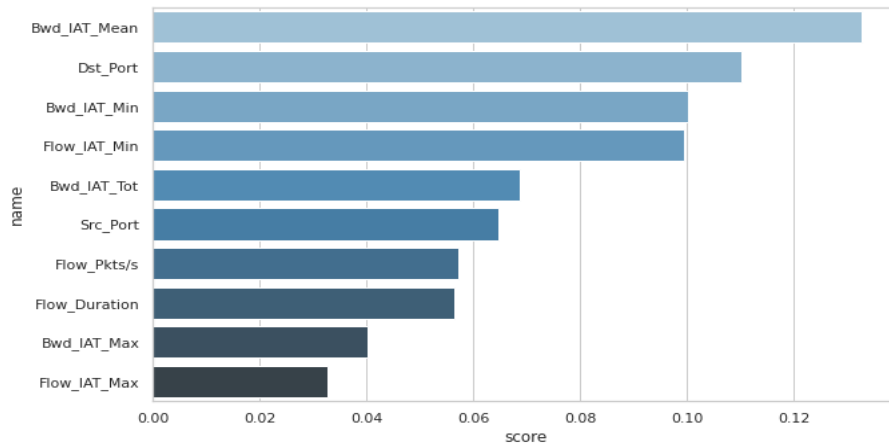


Figure 4.25: Feature Importance of Reconnaissance model

### 4.3.6 Category Theft

The performance of the models:

- Logistic Regression
  - Accuracy: 0.73
  - TPR: 0.73
  - FPR: 0.73
- SVM
  - Accuracy: 0.75
- RFT (tuned)
  - Accuracy: 0.97

The classification report of Random Forest Tree (RFT) model with hyperparameters tuned along with confusion matrix are shown Fig. 4.26 and Fig. 4.27. The model achieves a perfect score of 1 for recall of ‘Keylogging’ and for the precision of ‘Data\_Exfiltration’.

```

Classification Report

```

	precision	recall	f1-score	support
Keylogging	0.96	1.00	0.98	70
Data_Exfiltration	1.00	0.91	0.95	32
accuracy			0.97	102
macro avg	0.98	0.95	0.96	102
weighted avg	0.97	0.97	0.97	102

Figure 4.26: Classification report of Theft model

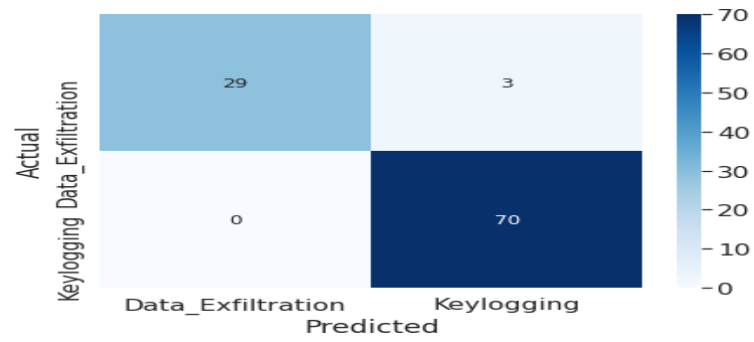


Figure 4.27: Confusion matrix of Theft model

The most influencing feature is ‘Src\_Port’ with a whopping score of 0.26. The next feature inline has a score of only 0.04 (Fig. 4.28). This score is the highest for any influencing feature for any model in stage 3.

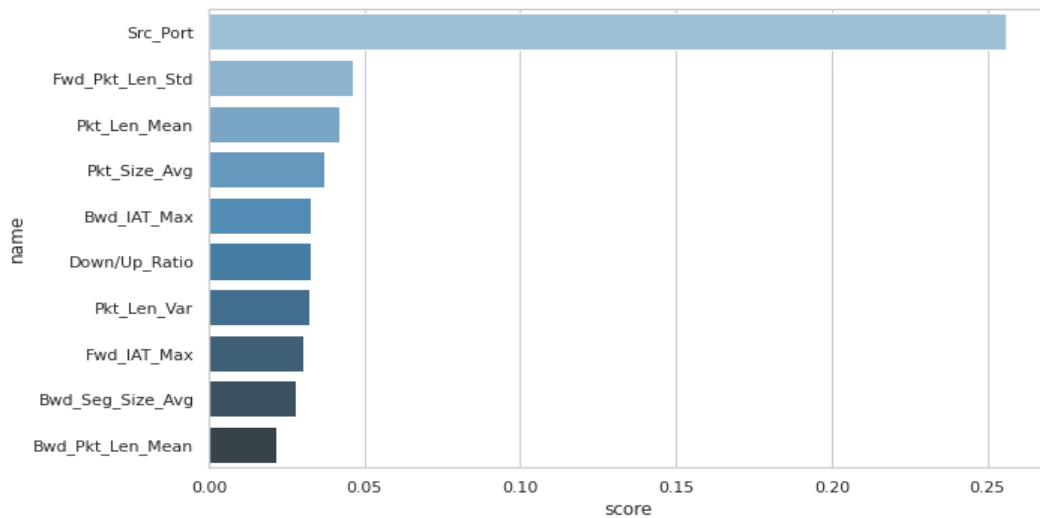


Figure 4.28: Feature Impotence of Theft model

Based on feature importance analysis of the stage 3 models, 10 most influencing features are selected, and models for categories “Reconnaissance” and “Theft” are trained only on the selected 10 features to further improve the performance of the models. The selected features are:

- ACK\_Flag\_Cnt
- Src\_Port
- Fwd\_Pkt\_Len\_Std
- Dst\_Port
- SYN\_Flag\_Cnt
- Fwd\_IAT\_Std
- Fwd\_Pkt\_Len\_Max
- Bwd\_IAT\_Std
- Bwd\_IAT\_Tot
- Bwd\_Pkt\_Len\_Std

The accuracy of model for category ‘Reconnaissance’ increase from 0.93 to 0.96. Likewise, the accuracy of the model for category ‘Theft’ improved from 0.97 to 0.99. The classification report of model ‘Theft’ is shown in Fig. 4.29.

Classification Report				
	precision	recall	f1-score	support
Keylogging	1.00	0.99	0.99	74
Data_Exfiltration	0.94	1.00	0.97	17
accuracy			0.99	91
macro avg	0.97	0.99	0.98	91
weighted avg	0.99	0.99	0.99	91

**Figure 4.29: Classification report of Theft model (top 10 influencing features)**

### 4.3.7 Category Mirari

The accuracy of the models:

- Logistic regression
  - Accuracy: 0.44
  - TPR: 0.44
  - FPR: 0.44
- Random forest Tree
  - Accuracy: 0.69

The classification report and influencing features are shown in Fig. 4.30 and Fig. 4.31. The features correlate weakly with the target classes hence the target matrix i.e. F1 score is lower compared to other models in stage 3. The most influencing feature ‘Protocol’ have a score of only 0.07 which indicates that the model is unable to map features on the target classes excluding sub-category ‘Mirari-Hostbrutefroceg’.

Classification Report				
	precision	recall	f1-score	support
Mirai-UDP Flooding	0.82	0.76	0.79	36640
Mirai-Hostbrutefroceg	0.84	0.98	0.90	24199
Mirai-HTTP Flooding	0.30	0.32	0.31	11235
Mirai-Ackflooding	0.27	0.21	0.24	10964
accuracy			0.69	83038
macro avg	0.56	0.57	0.56	83038
weighted avg	0.68	0.69	0.69	83038

Figure 4.30: Classification report of Mirari model



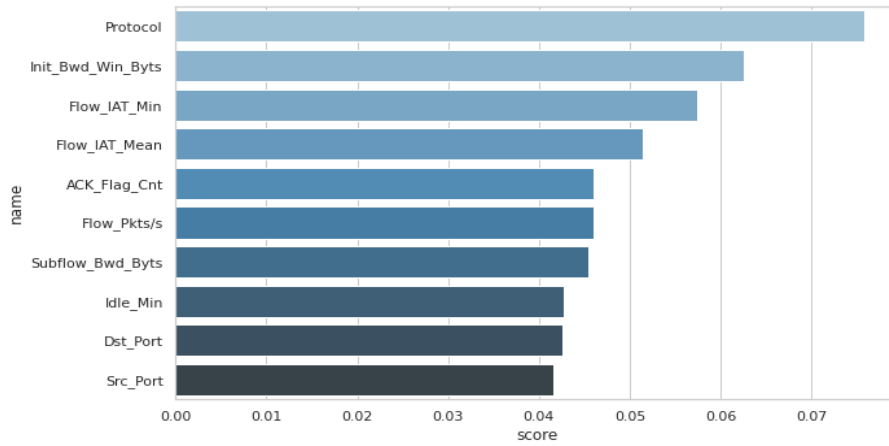


Figure 4.31: Feature importance of Mirari model

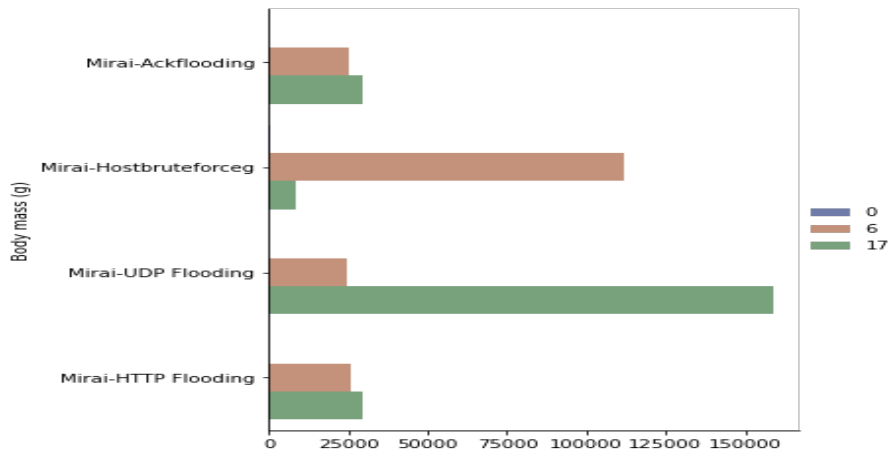


Figure 4.32: Class distribution on most influencing feature

To model’s below-par performance can be seen in the box plot. The subcategories are on y-axis, count on the y-axis and each box represents a nominal protocol value (0,6,17). The subcategory ‘Mirai-Ackflooding’ have the lowest F1 score which is due to the fact that for this subcategory, both protocol values 6 and 17 have roughly 27k instances. Therefore, the model is unable to extract useful patterns from the feature. Same case for sub-category ‘Mirari-HTTP Flooding’.

#### 4.3.8 Category Scan

The performance of the models:

- Logistic Regression
  - Accuracy: 0.70
  - TPR: 0.70
  - FPR: 0.70
- SVM
  - Accuracy: 0.71
- RFT
  - Accuracy: 0.86

The classification report in Fig. 4.33 shows F1 score for ‘Scan Host-port’ is 0.71 which improved from 0.08 in combined model. This can be visualize in the influencing features plot (as shown in Fig. 4.34) which is dominated by ‘Src\_Port’ and ‘Dst\_Port’ with scores of 0.40 and 0.23. The sub classes of the category ‘Scan’ interact uniquely with the features as no other subclass of either of the dataset have strong correlation with features.

Classification Report				
	precision	recall	f1-score	support
Scan Port OS	0.85	0.97	0.91	10532
Scan Hostport	0.91	0.58	0.71	4466
accuracy			0.86	14998
macro avg	0.88	0.78	0.81	14998
weighted avg	0.86	0.86	0.85	14998

Figure 4.33: Classification report of Scan model

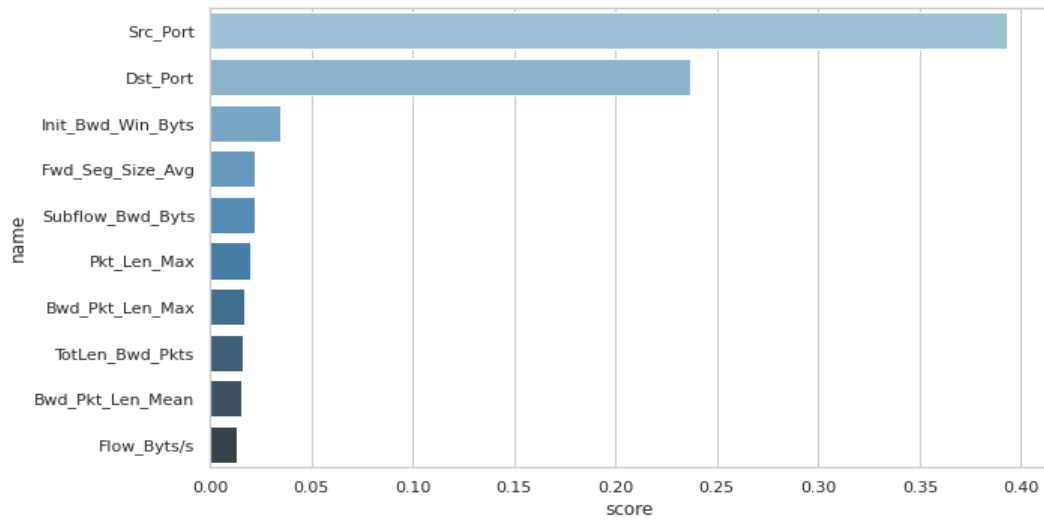


Figure 4.34: Feature importance of Scan model

Categories ‘DoS’ and ‘MTM ARP Spoofing’ have only one sub-category therefore they will not be further classified on stage 3. Table 4.2 displays the overall summary of category-specific models.

**Table 4.2:** Stage 3 models performance summary

Sub-Category	Models (Best Performing)	Accuracy	2 most influencing
DDoS	RFT (tuned)	1.00	Pkt_len_mean, Init_Bwd_Win_Byts
DoS (system 1)	RFT (tuned)	1.00	Pkt_len_Max, Pkt_Len_Mean
Reconnaissance	RFT (tuned) 10 Fea	0.96	Bwd_Iat_Mean, Dst_Port
Theft	RFT (tuned) 10 Fea	0.99	Src_Port, Fwd_Pkt_Len_Std
Mirari	RFT (tuned)	0.69	Protocol, Init_Bwd_Win_Byts
Scan	RFT (tuned)	0.86	Src_Port, Dst_Port
DoS (system 2)	No model	NA	NA
MTM ARP	No model	NA	NA

Variation in influencing features for each model of stage 3. This explains why the model trained on combine dataset unable to map features on target variable. Each category interacts with features differently therefore, by training models only on categories allowed the models to avoid underfitting specifically on minority classes as in this case the influence of the most occurring classes is avoided. This procedure also removed class im-balancing issues as for a specific category, the number of instances for all proceeding sub-categories are in similar range. This is another critical factor which hampers the performance of the model on minority classes as oversampling did not yield any improvement in the performance (on combined dataset). Hence by training category specific models, the issue of

un-balanced and different way each category interact with features is overcome.

# Transfer Learning

Both datasets are converted into images to feed into a CNN model. Initially, each row of the dataset is converted into a higher dimension i.e. 50x50x3 using DeepInsight. The converted data is shown in the Fig. 5.1.

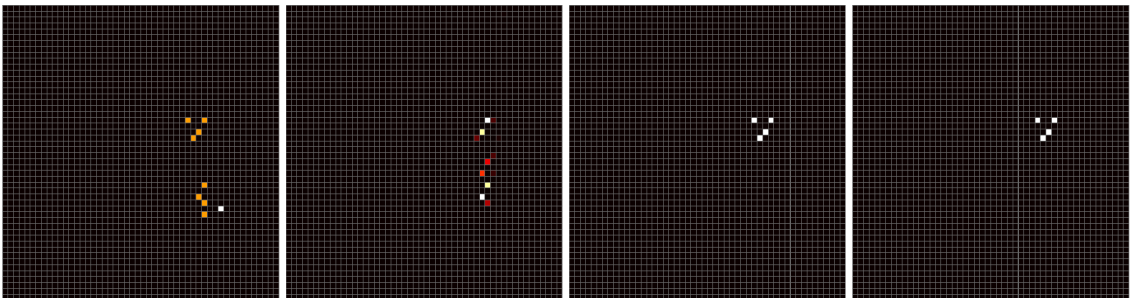


Figure 5.1: Image Data

VGG16 model is loaded with pre-trained weights frozen (hidden layers) and with the output layer removed. Some additional layers are attached on top of models which will be used in training on the converted dataset.

The class-wise breakdown of classification metric scores is shown in Fig. 5.2.

VGG16 Model Classification Report:				
	precision	recall	f1-score	support
0	0.00	0.00	0.00	185
1	0.95	1.00	0.98	3696
accuracy			0.95	3881
macro avg	0.48	0.50	0.49	3881
weighted avg	0.91	0.95	0.93	3881

**Figure 5.2: Classification report of model**

The accuracy of the model is 0.95 which is due to the model performance of the majority class. The model is predicting each sample in the test dataset as an anomaly. The model failed to extract any useful feature representation from the training data. Due to the unique nature of the dataset, the whole VGG16 model is trained (this time the hidden layers are not frozen) on MS Azure with higher computing resources but the model is still unable to extract any feature representation from the dataset to distinguish between target classes. This can be visualized in the converted images shown above. There is no pattern in the images which could be learned by the model.

The issue is resolved by converting each row into 8x8x3 RGB image (72 features). 72 features are converted into 162 pixels. The selection of the lower dimension of the image is inspired by the number of features. Using lower dimension assist the model to extract meaningful feature representations in convolution layers. This can also be visualized in Fig. 5.3.

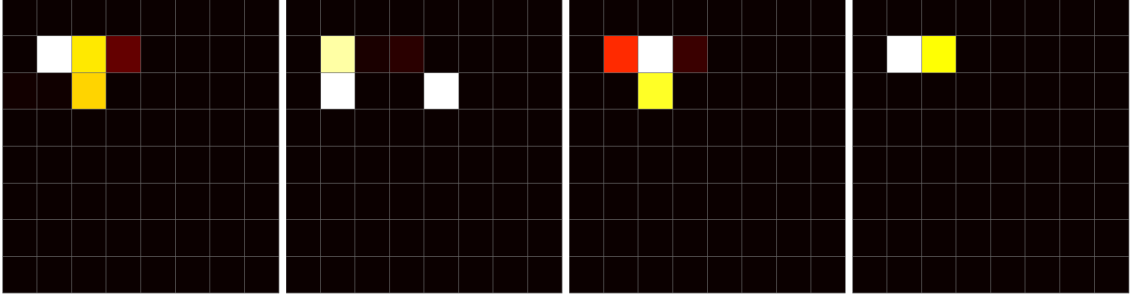


Figure 5.3: Image Date in 8x8x3 dimmensions

Training model on 8x8x3 yields best result. Additionally, equal number of examples from both classes i.e. Normal and Anomaly are selected and converted into images.

Table 5.1: Samples of each class

	Base Dataset	Target Dataset
Dataset	1	2
Number of examples	180k	30k
Dimensions	8x8x3	8x8x3

The sample of converted images of dataset 1 are shown in Fig. 5.3. The target column is label which have two classes i.e. Normal and Anomaly. Note that for class anomaly the converted image will varies as it is further categories and sub-categories. This part of project only focus on the ‘Label’.

## 5.1 Base Model

A shallow CNN model is trained on the converted dataset. Lower dimensions of the images limit the number of hidden layers in the model. Batch size is 32 and epochs are 15.



The classification report of base model is shown in Fig. 5.4.

CNN Gray Scale Model Classification Report:				
	precision	recall	f1-score	support
0	0.71	0.90	0.79	18140
1	0.86	0.63	0.73	17860
accuracy			0.77	36000
macro avg	0.79	0.76	0.76	36000
weighted avg	0.78	0.77	0.76	36000

Figure 5.4: Classification report of base model

And the train-validation accuracy curves of base model are shown in Fig. 5.5.

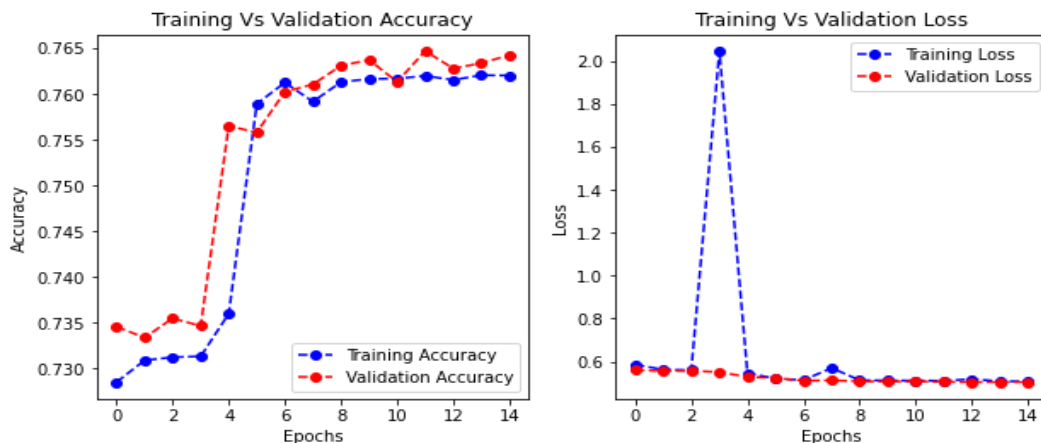


Figure 5.5: Base model training

## 5.2 Transfer Learning Model

The base model is loaded with the output layer removed. The hidden layers of the base model are frozen so that the weight information doesn't change during training. A few additional layers are attached

which will be trained on dataset 2. With similar setting, the transfer learning model is trained on dataset 2. Only 60k rows are used which is 1/3 of the sample used for training base model. The accuracy of the model improved from 0.77 to 0.90. The breakdown of evaluation metrics for transfer learning model are shown in Fig. 5.6.

Transfer Learning Model Classification Report:				
	precision	recall	f1-score	support
0	0.85	0.97	0.91	5999
1	0.97	0.83	0.89	6001
accuracy			0.90	12000
macro avg	0.91	0.90	0.90	12000
weighted avg	0.91	0.90	0.90	12000

Figure 5.6: Classification report of TL model

Whereas, the respective train-validation curves for transfer learning models can be observed in Fig. 5.7.

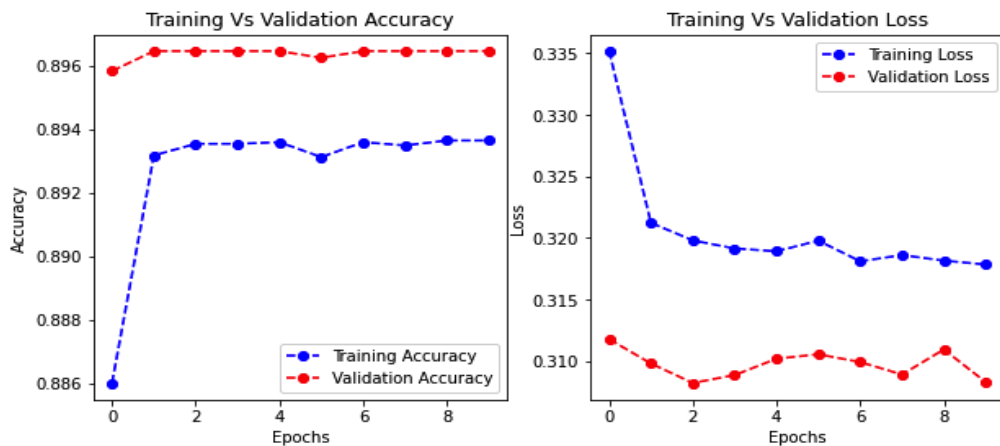


Figure 5.7: TL model training

Transfer learning help in improving the performance of the model if two datasets have similar feature space.

# Conclusion and Future Work

## 6.1 Conclusion

Intrusion detection systems (IDS) plays vital role in securing modern IoT infrastructure. This research focus improving the performance of the machine learning models by classifying each label feature at different stage and using feature engineering and feature selection to make model robust in first part. In second part the research focus on utilizing existing dataset to improve the performance of the model on sparse dataset if the feature space is similar by first converting tabular data into images and then using transfer learning using deep learning models.

## 6.2 Future Work

This research shows the importance of feature selection in improving the performance of the models. A more data centric approach i.e. feature interaction and dimensionality reduction have possibility of

further improving the performance of the models.

# References

- [1] Ge, Y., Yue, D. and Chen, L., 2017, November. Prediction of wind turbine blades icing based on MBK-SMOTE and random forest in imbalanced data set. In 2017 IEEE Conference on Energy Internet and Energy System Integration (EI2) (pp. 1-6). IEEE.
- [2] Sharma, A., Vans, E., Shigemizu, D., Boroevich, K.A. and Tsunoda, T., 2019. DeepInsight: A methodology to transform a non-image data to an image for convolution neural network architecture. *Scientific reports*, 9(1), pp.1-7.
- [3] Sugi, S.S.S. and Ratna, S.R., 2020, December. Investigation of machine learning techniques in intrusion detection system for IoT network. In 2020 3rd International Conference on Intelligent Sustainable Systems (ICISS) (pp. 1164-1167). IEEE.
- [4] Diro, A.A. and Chilamkurti, N., 2018. Distributed attack detection scheme using deep learning approach for Internet of Things. *Future Generation Computer Systems*, 82, pp.761-768.
- [5] Lopez-Martin, M., Carro, B., Sanchez-Esguevillas, A. and Lloret, J., 2017. Conditional variational autoencoder for pre-

- diction and feature recovery applied to intrusion detection in iot. *Sensors*, 17(9), p.1967.
- [6] Anthi, E., Williams, L., Słowińska, M., Theodorakopoulos, G. and Burnap, P., 2019. A supervised intrusion detection system for smart home IoT devices. *IEEE Internet of Things Journal*, 6(5), pp.9042-9053.
- [7] Amouri, A., Alaparthi, V.T. and Morgera, S.D., 2018, April. Cross layer-based intrusion detection based on network behavior for IoT. In 2018 IEEE 19th Wireless and Microwave Technology Conference (WAMICON) (pp. 1-4). IEEE.
- [8] Doshi, R., Apthorpe, N. and Feamster, N., 2018, May. Machine learning ddos detection for consumer internet of things devices. In 2018 IEEE Security and Privacy Workshops (SPW) (pp. 29-35). IEEE.
- [9] Shukla, P., 2017, September. ML-IDS: A machine learning approach to detect wormhole attacks in Internet of Things. In 2017 Intelligent Systems Conference (IntelliSys) (pp. 234-240). IEEE.
- [10] Ullah, I. and Mahmoud, Q.H., 2020. A two-level flow-based anomalous activity detection system for IoT networks. *Electronics*, 9(3), p.530.
- [11] Ullah, I. and Mahmoud, Q.H., 2019, January. A two-level hybrid model for anomalous activity detection in IoT networks. In 2019 16th IEEE Annual Consumer Communications Networking Conference (CCNC) (pp. 1-6). IEEE.

- [12] Alalade, E.D., 2020, June. Intrusion detection system in smart home network using artificial immune system and extreme learning machine hybrid approach. In 2020 IEEE 6th World Forum on Internet of Things (WF-IoT) (pp. 1-2). IEEE.
- [13] Pamukov, M.E. and Poulkov, V.K., 2017, September. Multiple negative selection algorithm: Improving detection error rates in IoT intrusion detection systems. In 2017 9th IEEE international conference on intelligent data acquisition and advanced computing systems: technology and applications (IDAACS) (Vol. 1, pp. 543-547). IEEE.
- [14] Ullah, I. and Mahmoud, Q.H., 2020, October. A technique for generating a botnet dataset for anomalous activity detection in IoT networks. In 2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC) (pp. 134-140). IEEE.
- [15] Ullah, I. and Mahmoud, Q.H., 2020, May. A scheme for generating a dataset for anomalous activity detection in iot networks. In Canadian Conference on Artificial Intelligence (pp. 508-520). Springer, Cham.
- [16] Arik, S.Ö. and Pfister, T., 2021, May. Tabnet: Attentive interpretable tabular learning. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 35, No. 8, pp. 6679-6687).
- [17] Borisov, V., Leemann, T., Seßler, K., Haug, J., Pawelczyk, M. and Kasneci, G., 2021. Deep neural networks and tabular data: A survey. arXiv preprint arXiv:2110.01889.

- [18] Hussain, F., Abbas, S.G., Husnain, M., Fayyaz, U.U., Shahzad, F. and Shah, G.A., 2020, November. IoT DoS and DDoS attack detection using ResNet. In 2020 IEEE 23rd International Multitopic Conference (INMIC) (pp. 1-6). IEEE.
- [19] Sun, B., Yang, L., Zhang, W., Lin, M., Dong, P., Young, C. and Dong, J., SuperTML: Domain Transfer from Computer Vision to Structured Tabular Data through Two-Dimensional Word Embedding.
- [20] Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H. and He, Q., 2020. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1), pp.43-76.
- [21] Yilmaz, S., Aydogan, E. and Sen, S., 2021. A Transfer Learning Approach for Securing Resource-Constrained IoT Devices. *IEEE Transactions on Information Forensics and Security*, 16, pp.4405-4418.
- [22] Chiba, S. and Sasaoka, H., 2021, May. Basic Study for Transfer Learning for Autonomous Driving in Car Race of Model Car. In 2021 6th International Conference on Business and Industrial Research (ICBIR) (pp. 138-141). IEEE.
- [23] Nalini, M.K. and Radhika, K.R., 2020, October. Comparative analysis of deep network models through transfer learning. In 2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC) (pp. 1007-1012). IEEE.



- [24] Liu, Q., Cheng, L., Ozcelebi, T., Murphy, J. and Lukkien, J., 2019, May. Deep reinforcement learning for IoT network dynamic clustering in edge computing. In 2019 19th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID) (pp. 600-603). IEEE.
- [25] Nagisetty, A. and Gupta, G.P., 2019, March. Framework for detection of malicious activities in IoT networks using keras deep learning library. In 2019 3rd international conference on computing methodologies and communication (ICCMC) (pp. 633-637). IEEE.
- [26] Shao, L., Zhu, F. and Li, X., 2014. Transfer learning for visual categorization: A survey. IEEE transactions on neural networks and learning systems, 26(5), pp.1019-1034.
- [27] Phan, T.V., Sultana, S., Nguyen, T.G. and Bauschert, T., 2020, February. Q-TRANSFER: A Novel Framework for Efficient Deep Transfer Learning in Networking. In 2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIIC) (pp. 146-151). IEEE.