Encoded Traffic Analysis with Entropy Evaluation and Neural Networks

A PROJECT REPORT

Submitted by

Harsh J Shah [Reg No: RA1711020010131]

Prithish Ghosh [Reg No: RA1711020010165]

Under the Guidance of

Dr. Usha G

(Assistant Professor, Department of Software Engineering)

In partial fulfilment of the Requirements for the Degree

Of

BACHELOR OF TECHNOLOGY

DEPARTMENT OF SOFTWARE ENGINEERING

FACULTY OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR – 603203

May 2021

BONAFIDE CERTIFICATE

I certify that this project report, entitled **Encoded Traffic Analysis with entropy evaluation and Neural Networks**, is the real work of "Harsh J Shah [REG NO: RA1711020010131], Prithish Ghosh [REG NO: RA1711020010165]" who is in charge of the project work was under my supervision. I also confirm that the work reported here is, to the best of my knowledge and belief, not part of another work or dissertation on the basis of which this or another candidate was previously awarded a title or award.

Dr. Usha G	Dr. C Lakshmi
GUIDE	**HEAD OF THE DEPARTMENT**
Assistant Professor	
Dept. of Software Engineering	Dept. of Software Engineering

Signature of Internal Examiner **Signature of External Examiner**

ACKNOWLEDGEMENT

We take this opportunity to thank Dr. MUTHAMIZHCHELVAN, Director (E&T) of the SRM Institute for Science and Technology, for providing the necessary infrastructure for the development of the project and the software engineering department at SRM IST for creating an environment in which to work on the project. We would like to thank Dr. Uma (Assistant Professor, SG) and Ms. Jayasudha (Assistant Professor, OG), our project coordinators, for their valuable support during the entire project development process. We would like to thank our guide, Dr.Usha G (Assistant Professor, Mr. G) for her valuable support, constant encouragement, personal attention, quick participation and providing an excellent research environment. Despite her busy schedule, she supported us all week. Working to Complete the Project Finally, we thank everyone who participated in the incubation of this project.

Harsh J Shah

Prithish Ghosh

TABLE OF CONTENTS

Abstract

Through this project we are going to analyse the entropy estimation for encoded traffic categorization. The encoded traffic detector able to reason traffic in each way that (i.e., Encoded/decoded). The performance would be based mostly upon the network traces. The traffic detector that is targeted to control during a privacy protective atmosphere will be present. A customization for 2 totally different tasks will be performed, wherever unencrypted traffic is born and solely encrypted traffic is forwarded. The classification of the encoded detector is exclusively supported data gather from the primary packet of a flow. Header fields in addition because the payload is taken under consideration. The main notion of an encoded traffic detector is based on estimating the payload's entropy and comparing the retrieved value to the entropy of a typical dispersed payload. supported ground network traces with encoded traffic and real network traces it'll be shown that the encoded detector is ready to filter an outsized fraction of unencrypted traffic, whereas an outsized fraction of encrypted flows is forwarded.

CHAPTER 1

INTRODUCTION

During the final years there was sizable boom in programs which can be the use of the Internet for communication. Applications like P2P file sharing, VoIP communication, Video on Demand or videoconferencing are of developing interest. Due to limited community resources such programs are raising new wishes inside the Internet. Several techniques had been proposed to higher aid such programs, those techniques are summarized beneath neath the term Quality of Service (QoS) for the Internet”. QoS helps offerings which have precise qualitative wishes by supplying them with the right quality. To offer QoS to those programs the visitors belonging to those programs has to be recognized inside a multiplexed visitors aggregate. This Identification of visitors is referred to as the visitor’s category. Traffic category is similarly needed to apprehend the community behaviour, that's an crucial prerequisite for community operation. Based on a stepped forward understanding, for instance community protocols are improved or a higher community control is implemented. A crucial style withinside the Internet is that regulations evolve into digital life. During the final years the regulations has enacted a few criminal tips which enlarged the right for privacy to the arena of computer networks. Content organizations and Internet Service Providers must appreciate and defend the privacy of their clients. Consequently, new architectures and algorithms want to be advanced to help network operation according with user privacy. Especially the research situation of passive network monitoring is advocated by those criminal tips, as traffic from real clients are accumulated in passive monitoring. Legacy network tracking programs are analysing all patron visitors to carry out their algorithms. These programs must be changed to carry out their obligations on a pre-filtered subset of the unique visitors. The pre-filtering needs to be implemented as “close” to the amassing point as possible. This undertaking specializes in a visitor’s classifier f1 rating this is able to classify visitors into encoded and unencoded visitors.

CHAPTER 2

PROJECT OVERVIEW

2.1 LITERATURE SURVEY

Title	Publication	Author	Year	Techniques
Capture Network Traffic Using Deep Neural Networks	IEEE	Dimitra Chamou Petros Toupas	2019	ML techniques, traffic classification techniques.
Real Network Traffic Collection and Deep Learning for Mobile App Identification	Hindawi	Marco Picone	2020	DL techniques to APP-ID techniques.
Network Traffic Analysis using Big Data and Deep Learning Techniques	IEEE	Nabil Hmina Habiba Chaoui	2020	Traffic classification techniques, DL techniques
Practical evaluation of encrypted traffic classification based on a combined method of entropy estimation and neural networks	ETRI	Kun Zhou Teng Hu Chenhuang Wu	2020	Encrypted art of deep learning technique, machine learning technique
A Survey of Methods for Encrypted Traffic Classification and Analysis	Wiley InterScience	John Wiley	2014	Classification technique, graphical technique, payload inspection technique

2.2 PROBLEM DESCRIPTION

As network data encoding becomes more common, encoded traffic classification plays an increasingly significant role in cybersecurity. First, we'll quickly go through three traffic encoding mechanisms: IPsec, SSL/TLS, and TLS. We'll use entropy estimation and neural networks to evaluate the performance of support vector machines and logistic regression for traffic analysis. Second, using entropy analysis, network traffic is classified as encoded or plaintext. Any categorised mistreatment neural networks are then encoded traffic. We will correct the improved being by this analysis.

2.3 REQUIREMENTS GATHERINGS

● Prototyping: Specifications for prototype consists of preparing and then creating a model that can be provided for consumers to "play," check and change.

● Interactions: Interviews are the primary means of gathering information where the program manager can meet face-to-face with specific clients or subject matter specialists Interviews are the primary means of gathering information where the program is located.

● Focus Groups: A synergistic conversation is taking place between individuals who are representations of consumers or buyers in relation to the requirements of the company.

2.4 REQUIREMENT ANALYSIS

2.4.1 Functional requirements

- LAN / WAN connectivity: LAN/ WAN connectivity should be in the proper way.
- Available bandwidth: Bandwidth is an important component in order to work efficiently.

- Server and workstation distribution: Proper server distribution and wide range is important.
- Application restriction: We should use only selective applications and capture those data.
- Sensitive data availability to select groups: we should use only selective data.
- Encoded data for Mobile users: For mobile useability we use the encoded data for capturing purpose.

2.4.2 Non-Functional requirements

- Accuracy: The output provided by the algorithms must be consistent & accurate.
- Performance: The trained model must be able to complete the task with ease and speed.
- Security: The input dataset and output should be secure and integrity must be maintained to prevent tampering data.
- Stability: The application must be able to work under a stressful environment with stability.

2.5 DATA SOURCE

The data collected has been captured by real time messages transferred between different mobile devices. The network traffic was captured not only for any one third party app but it consists of multiple third-party apps data.

The data captured from different third-party apps during real time network capturing is been used in this project for entropy evaluation.

2.6 COST ESTIMATION

In the practical ways for reaching immediate goals, we will apply the heuristic technique for solving issues, learning, or finding.

$$E = a(KLOC)^b$$

$$time = c(Effort)^d$$

$$Personrequired = Effort/time$$

COCOMO (Constructive Cost Model) is a regression version primarily based totally at the wide variety of strains of code (LOC) & good predictor of time, effort, cost, and quality.

Software Projects	a	b	c	d
Organic	2.4	1.05	2.5	0.38
Semi Detached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

E – Effort applied
KLOC – estimated number of lines of code (expressed in thousands)

Cost:

•So considering we may have 500 to 800 lines of code:

$$E = a(KLOC)^b$$

E= (2.4)(500/1000)(1.05)

E= 1.26

D= (2.5)(1.26)(0.38)

D= 1.187

N= E/D

N= 1.061 Persons

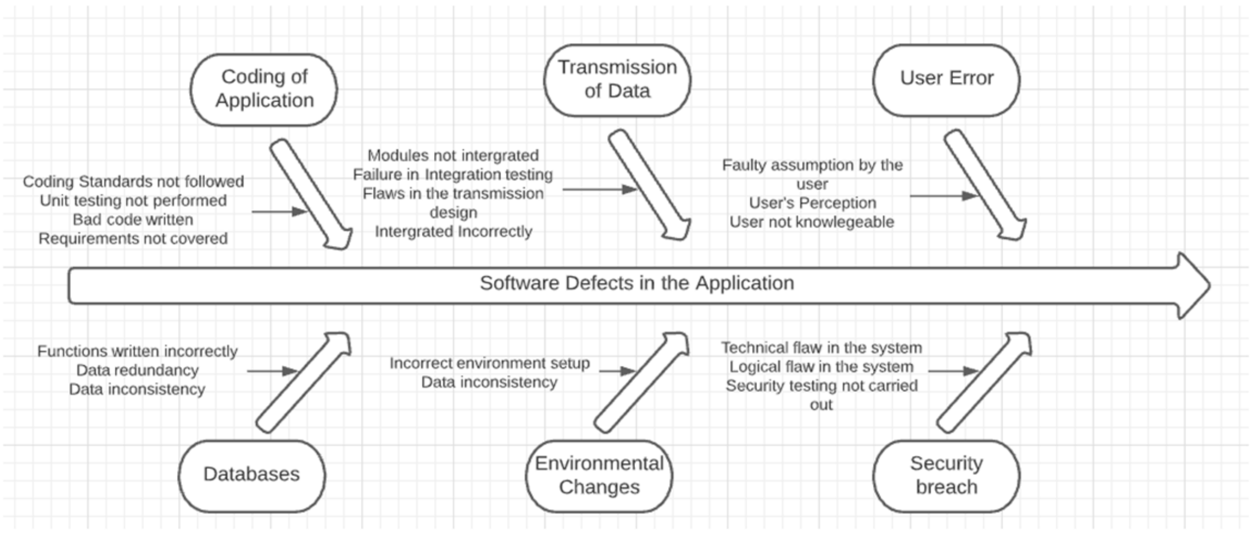
So, if we need to complete our work before time, we can assign less than or equal to 2 people to do it.

2.7 PROJECT SCHEDULE

Time duration	**Checkpoints for project**
10.12.2020-16.12.2020	Topic discussion
17.12.2020-22.12.2020	Feasibility selected topics
23.12.20-31.12.20	Requirement’s gatherings
4.1.21-11.1.21	literature survey and findings
12.1.21-23.1.21	cost estimation and model selection
25.1.21-10.2.21	System architecture diagram findings
11.2.21-26.2.21	UML diagrams verification
3.3.21-10.3.21	Relational model
11.3.21-22.3.21	Implementation
23.3.21-31.3.21	ER diagram
5.3.21-14.4.21	SWE unit testing
15.4.21-19.4.21	module description
20.4.21-26.4.21	Defect analysis ,cost, performance
27.04.21-3.05.21	Conclusion

2.8 RISK ANALYSIS:

RISK IDENTIFIED	PROBABILITY OF RISK OCCURRENCE
Bad annotation of training and testing datasets.	Very unlikely to occur(0-10%)
Poor algorithm assumptions and parameters.	unlikely to occur(11-40%)
Preferred data filtering process.	May occur about half of the time(41-60%)
Failure to understand objectives not understanding the data.	Likely to occur(61-70%)
Avoid leaks example : information, not enough filtering process, not enough data to rectify entropy system for neural networks.	Likely to occur(71-90%)


Coding of Application
Transmission of Data
User Error
Coding Standards not followed
Unit testing not performed
Bad code written
Requirements not covered
Modules not intergrated
Failure in Integration testing
Flaws in the transmission design
Intergrated Incorrectly
Faulty assumption by the user
User's Perception
User not knowlegeable
Software Defects in the Application
Functions written incorrectly
Data redundancy
Data inconsistency
Incorrect environment setup
Data inconsistency
Technical flaw in the system
Logical flaw in the system
Security testing not carried out
Databases
Environmental Changes
Security breach

2.9 SOFTWARE REQUIREMENT SPECIFICATION

1.Problem Description:

As network data encoding becomes more common, encoded traffic classification plays an increasingly significant role in cybersecurity. First, we'll quickly go through three traffic encoding mechanisms: IPsec, SSL/TLS, and TLS. We'll use entropy estimation and neural networks to evaluate the performance of support vector machines and logistic regression for traffic analysis. Second, using entropy analysis, network traffic is classified as encoded or plaintext. Any categorised mistreatment neural networks are then encoded traffic. We will correct the improved being by this analysis.

1.1. Introduction

1.1.1. Aim:

The project's goal is to show that encoded traffic analysis may be a useful tool for network administrators and security researchers, as well as to identify and explain the most serious encoded traffic analysis-based threats. As is commonly the case in cyber security, a tool is also used with and while not malicious intent. Through this project, we aim to present each aspect of encoded traffic analysis and process.

1.1.2 Document Convention:

This document uses Arial as font theme for normal content and times as font theme for sub-headings. Font size of content is 11 while subheadings have a font size of 14 and they are in bold. In addition, important text in the content is highlighted by bolding the text. Furthermore, the goals for higher-level specifications are believed to be inherited from the comprehensive specifications.

1.1.3 Product Scope:

The product is intended to use a network traffic analyser tool through which we can detect network traffic and capture the overall traffic with malicious activity. This product is to prevent most dangerous encoded traffic analysis-based attacks.

2.Description

2.1 Perspective:

This product helps to analyse network traffic.

Its main perspectives are:

● An open-source tool to detect and analyse network traffic.

●The captured data will be processed and the pattern of the messages transferred will be analysed using the neural networks.

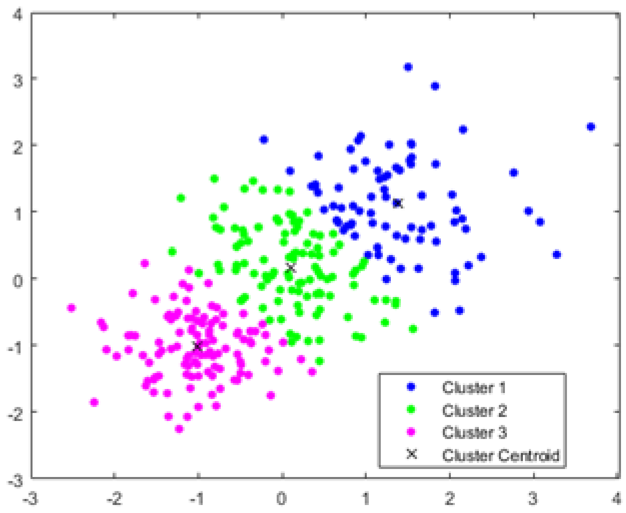
● Captured data will be used in the entropy system which will rectify us with the exact f1 score.

2.2 Product Function:

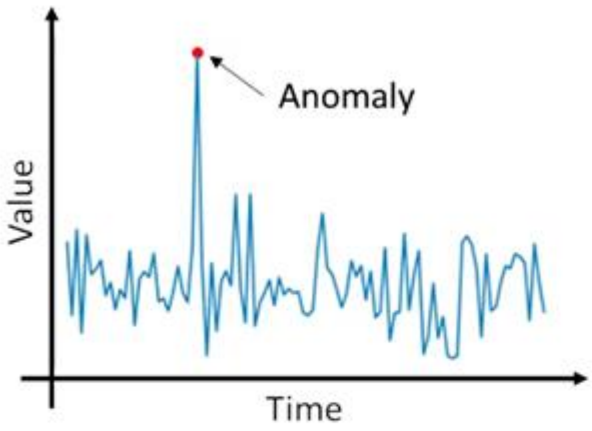
- Collecting Dataset: Creating a dataset for the training of the models is the primary task. Datasets can be created manually through captured network data and then augmented.
- Normalization: Datasets are normalized for filtering overall data.
- Pre-Processing: sending the dataset to the model, the dataset should be pre-processed. This includes splitting the data into several folds.
- Training Models: To do an optimal entropy evaluation process, they should be trained and validated with neural networks concepts (RNN, SVM).
- Detecting entropy evaluation: Training of models to detect accuracy of captured data from the pool of data consisting of legitimate and captured data.

Objective of Traffic analysis

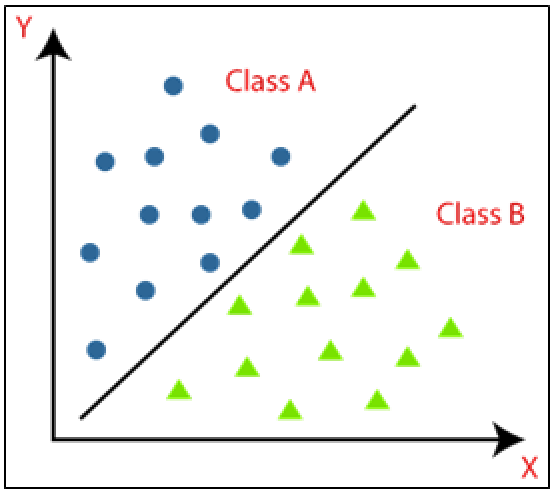
1. Clustering of Data



2. Anomaly Detection



3. Classification and Types



The Process

1. During the message transfer from one device to another over the internet, the encoded traffic data is captured by the tool named Wireshark to analyse the traffic.
2. Once the data has been fetched, we shall try to analyse and form patterns of the data using neural networks. The pattern hence formed has to be decoded and the output data communication between the two devices can be observed.
3. This will take a longer time to process the data of each message and find and match with the pattern, so to get proper and optimised work we require to calculate errors.

Through this process of error calculation and optimisation, output generation
data would provide accurate and faster results.

Project flow

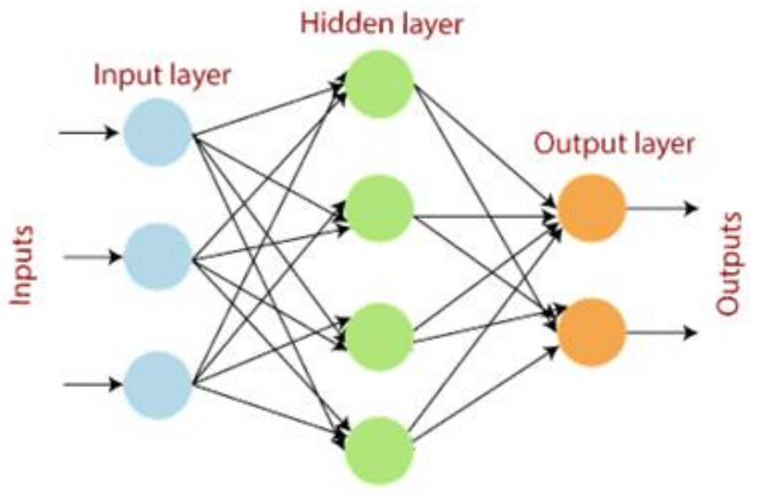
Now as we have the data which has been captured through Wireshark, the data is now available to be processed by using neural networks.
For processing the data, we will provide feature learning by using autoencoders to do this.

1. The input data is processed, the encoders using latent space and the data is reconstructed.

2. During this process of reconstruction of data using decoders there will be loss of data, which can be rectified using error calculation.

Why Autoencoders?

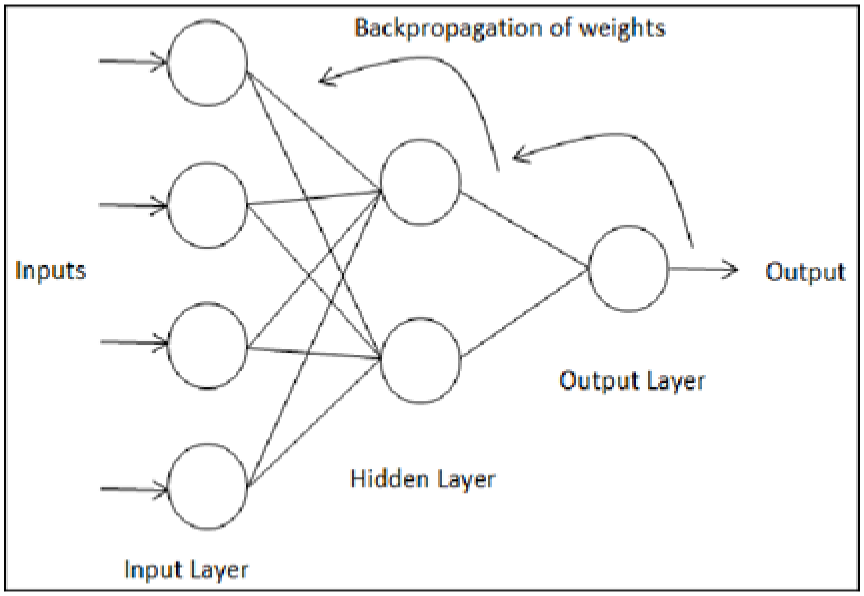
1. They compress the data into a lower-dimensional code, which they then use to rebuild the output.
2. They are a sort of feedforward neural network in which the input and output are the same.
3. Autoencoders are lossy in nature, i.e. the output does not perfectly mimic the input and therefore there is reconstruction error left out.



<u>Error calculation and optimization</u>

1. To find the error and the loss of data by autoencoders, backpropagation of weights technique is used.
2. In this process the backtracking of output data is done.
3. It goes back through history from the output layer to the concealed layer, then back to the input layer.
4. With this the difference is calculated and treated as error.

The resultant of error calculation is then added with the output data to find the pattern.



2.2.1 Characterization:

This software(tool) product can be used by whosoever wishes to be vigilant of the applications they access and to be sure that apps they access within Wi-Fi signals with minimum bandwidth.

2.2.2 Operating Environment:

This software tool would be available on PCs. Also, there would be a version available for the upcoming smart capturing process. For mobiles, this tool would be able to work on all platforms including android, windows, iOS as well.

2.2.3 Design implementation:

One of the biggest constraints would be that there would be constant requirement of internet connection like Wi-Fi networks or LAN connections with proper bandwidth to operate. As the product would be using a big dataset to analyse entropy, it might get slow and it might get tough to get quick responses. Reverse engineering of the product is not possible unless permitted by the company.

2.2.4 User Documentation:

There would be a user guide as well along with the product on instructions on how to install this tool on various platforms.

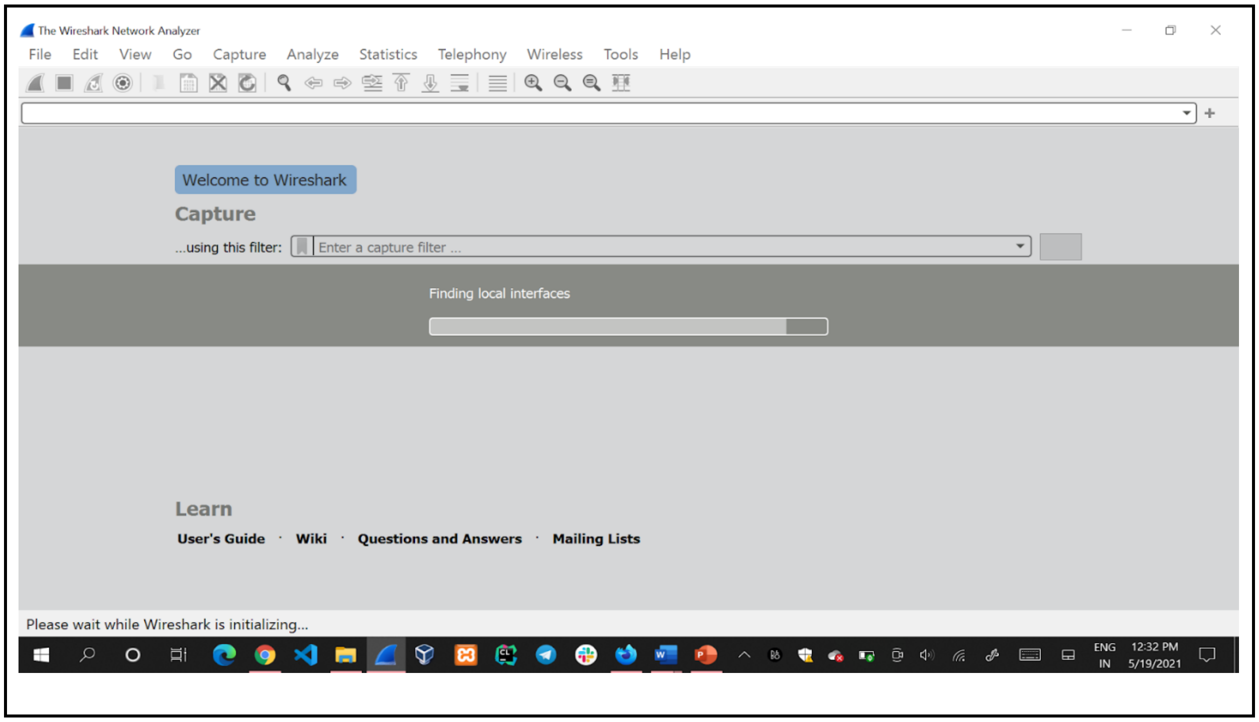
2.2.5 Assumption and Dependencies:

The project is to present encoded traffic analysis by tracking the network traffic between two devices. The captured data will be processed and the pattern of the messages transferred will be analysed using the neural networks.

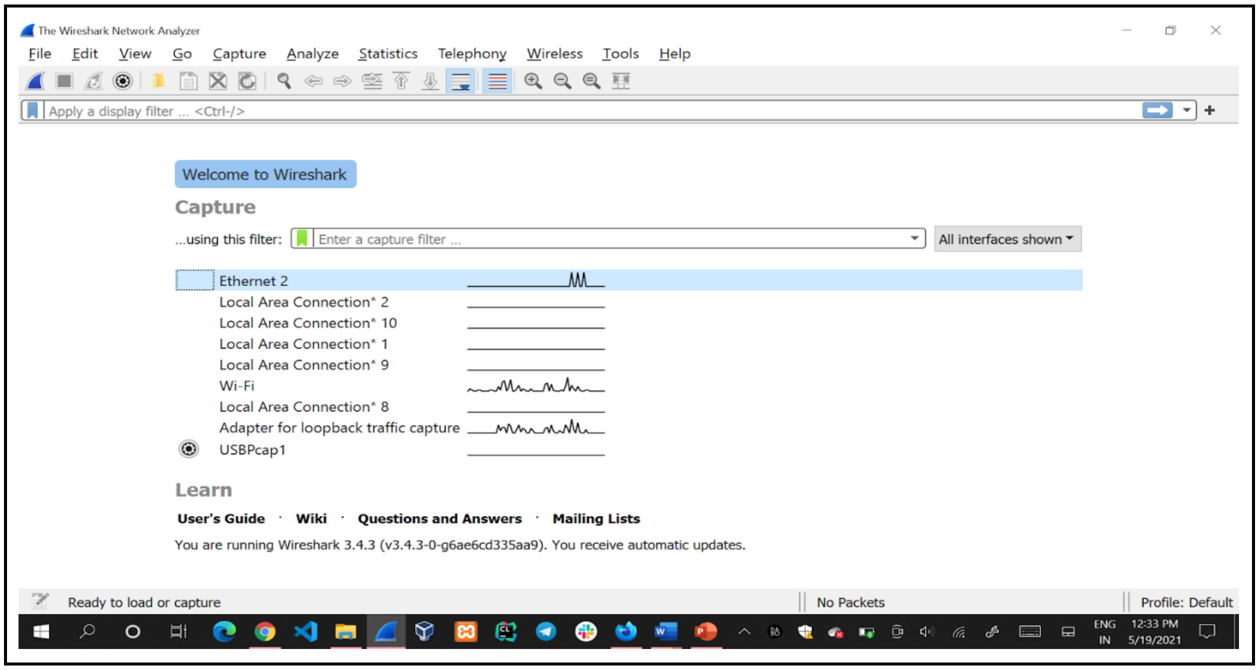
2.3 EXTERNAL REQUIREMENTS

2.3.1 UI

The UI of the desktop software would be very interactive and familiar to other networking applications. Background theme would be in white color. There would be a navigation bar from the left side where we would be having all our options and functions. Other than that, an bandwidth / wifi capture console would be a great interactive experience for users as they would be using easy to understand filters and automatically guiding search engines.



Please wait while Wireshark is initializing...


The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Welcome to Wireshark
Capture
...using this filter:
Enter a capture filter ...
All interfaces shown
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists
Ready to load or capture
No Packets
Profile: Default
ENG IN
12:33 PM 5/19/2021

Ethernet 2
Local Area Connection* 2
Local Area Connection* 10
Local Area Connection* 1
Local Area Connection* 9
Wi-Fi
Local Area Connection* 8
Adapter for loopback traffic capture
USBPcap1

You are running Wireshark 3.4.3 (v3.4.3-0-g6ae6cd335aa9). You receive automatic updates.

2.3.2 Hardware interfaces:

Hardware specifications are:

System: Laptop / Desktop with intel i3 processor

Operating system: Windows 7,8,10

Hard Drive: 240GB SSD

Ram: 8GB

Router Machine: TP-LINK with Broadband LAN connection preferred.

2.3.4 Software Interfaces:

Computer Specifications describe the concept of computing resource requirements and standards that need to be implemented on a device to ensure the optimum operation of the program. Software specifications for the project:

Operating system: Windows 10

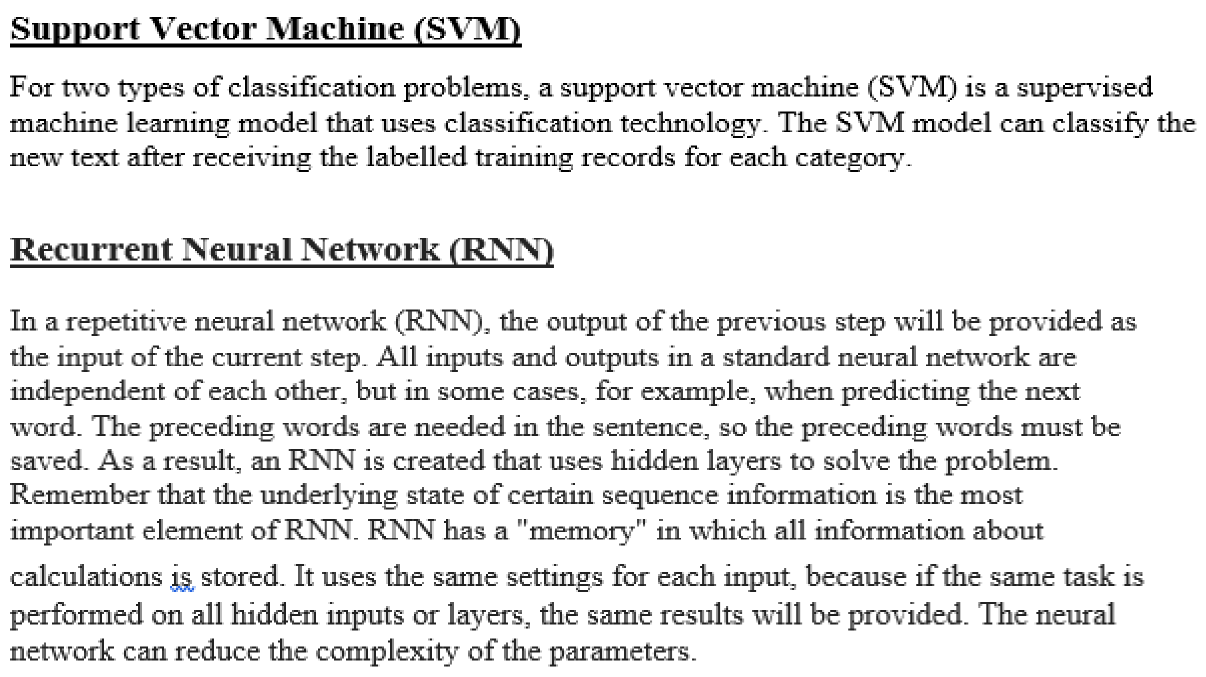
coding language: Python

2.3.4 Communication interfaces:

Since the different components of the system are responsible for different functions and as they are dependent on each other as well, then communication between them is important. All components would be looking for each other for their own data to be processed in some way. So, for this, the underlying operating system would be responsible and will carry out all the communication or data transfer processes.

2.4 SYSTEM FEATURES

1. Data collection
    1. As the mobile device is connected with the router and the router with the laptop device to capture data using Wireshark.
    2. The data is collected and stored in cloud storage for further process.
2. Entropy
    1. Once the data is captured, it can now be used by autoencoders to encode and decode the data.
    2. Autoencoders compress data into a lower-dimensional code, which is then used to reconstruct the output.
3. Neural
    1. Networks The data is now being processed and reconstructed using the neural networks schema.
    2. Through this a pattern is formed to be able to verify the data.
4. Analysis
    1. As the data is now completely ready, it has been matched with the pattern available.
    2. The proper analysis of the data is done using pattern matching.
    3. Thus after proper analysis the resultant analyzed data is given.



Support Vector Machine (SVM)

For two types of classification problems, a support vector machine (SVM) is a supervised machine learning model that uses classification technology. The SVM model can classify the new text after receiving the labelled training records for each category.

Recurrent Neural Network (RNN)

In a repetitive neural network (RNN), the output of the previous step will be provided as the input of the current step. All inputs and outputs in a standard neural network are independent of each other, but in some cases, for example, when predicting the next word. The preceding words are needed in the sentence, so the preceding words must be saved. As a result, an RNN is created that uses hidden layers to solve the problem. Remember that the underlying state of certain sequence information is the most important element of RNN. RNN has a "memory" in which all information about calculations is stored. It uses the same settings for each input, because if the same task is performed on all hidden inputs or layers, the same results will be provided. The neural network can reduce the complexity of the parameters.

2.5 OTHER NON-FUNCTIONAL REQUIREMENTS

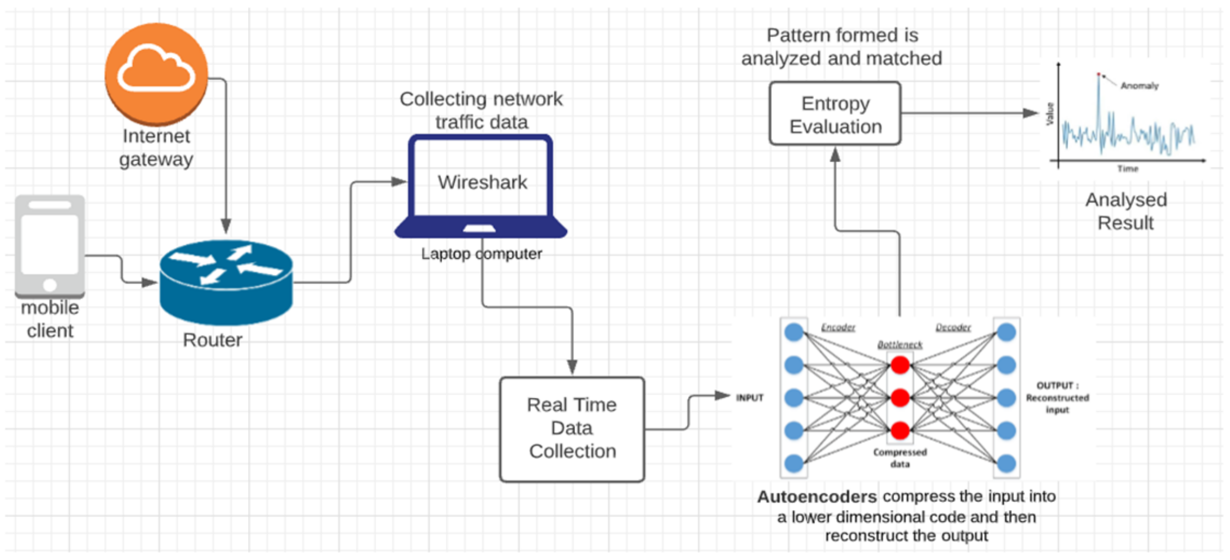
2.5.1 Accuracy – The output provided by the algorithms must be consistent and accurate.
2.5.2 Performance – The trained model must be able to complete the task with ease and speed.
2.5.3 Security – The input datasets and outputs must be secure and integrity must be maintained to prevent tampering of data.
2.5.4 Stability – The application must be able to work under a stressful environment with stability.

2.5.5 Business Rule:
This product is open source and available on google so in-order to use this tool needs one administrator who can manage the overall process as well the networking events.

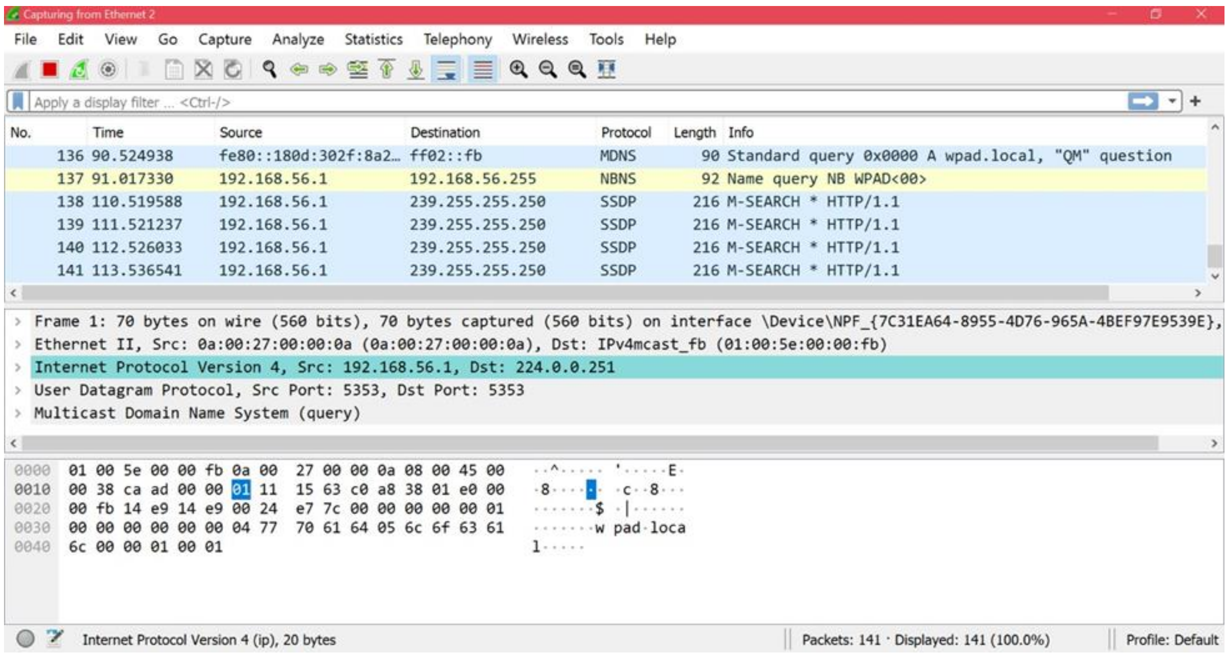
2.6 Other Requirements:
In initial stages, software is believed to take a lot of computation time, so it is recommended for users to have a high RAM to get the most out of the product.

3. ARCHITECTURE & DESIGN:

3.1 SYSTEM ARCHITECTURE:


Internet gateway
mobile client
Router
Collecting network traffic data
Wireshark
Laptop computer
Real Time Data Collection
INPUT
Encoder
Bottleneck
Decoder
Compressed data
OUTPUT : Reconstructed input
Autoencoders compress the input into a lower dimensional code and then reconstruct the output
Pattern formed is analyzed and matched
Entropy Evaluation
Anomaly
Value
Time
Analysed Result

3.2 INTERACTION PROTOTYPING (UI):


Capturing from Ethernet 2
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

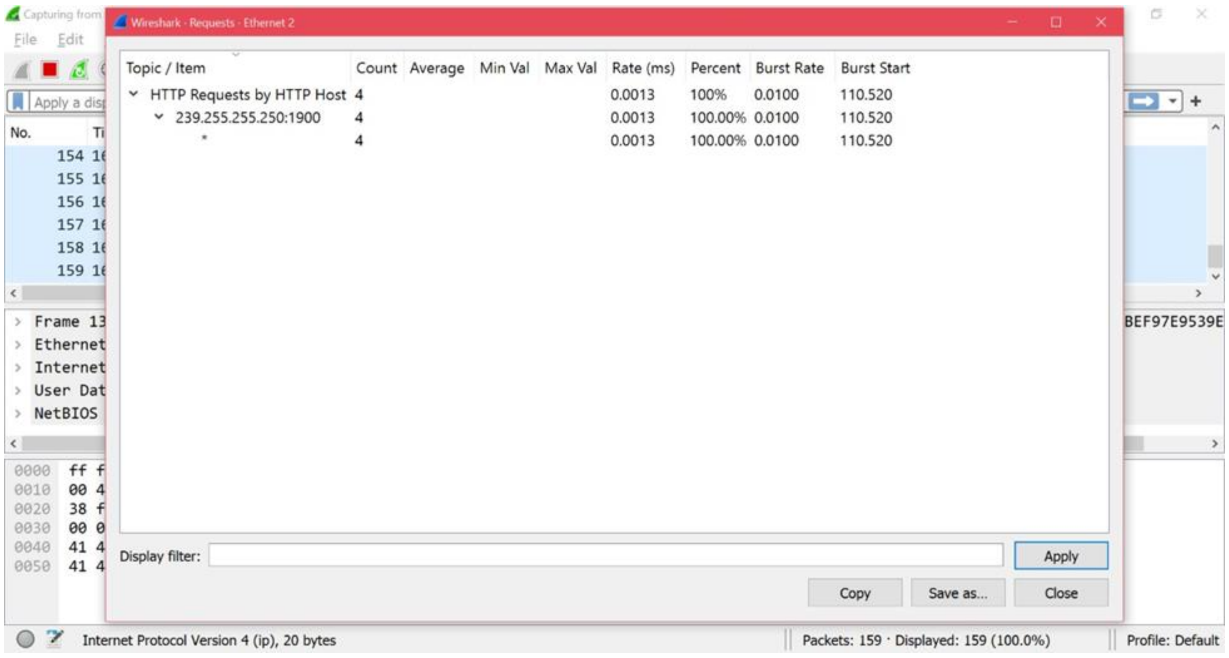
```
No.   Time        Source                  Destination       Protocol  Length  Info
  136 90.524938   fe80::180d:302f:8a2…    ff02::fb          MDNS          90  Standard query 0x0000 A wpad.local, "QM" question
  137 91.017330   192.168.56.1            192.168.56.255    NBNS          92  Name query NB WPAD<00>
  138 110.519588  192.168.56.1            239.255.255.250   SSDP         216  M-SEARCH * HTTP/1.1
  139 111.521237  192.168.56.1            239.255.255.250   SSDP         216  M-SEARCH * HTTP/1.1
  140 112.526033  192.168.56.1            239.255.255.250   SSDP         216  M-SEARCH * HTTP/1.1
  141 113.536541  192.168.56.1            239.255.255.250   SSDP         216  M-SEARCH * HTTP/1.1
```

```
> Frame 1: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface \Device\NPF_{7C31EA64-8955-4D76-965A-4BEF97E9539E},
> Ethernet II, Src: 0a:00:27:00:00:0a (0a:00:27:00:00:0a), Dst: IPv4mcast_fb (01:00:5e:00:00:fb)
> Internet Protocol Version 4, Src: 192.168.56.1, Dst: 224.0.0.251
> User Datagram Protocol, Src Port: 5353, Dst Port: 5353
> Multicast Domain Name System (query)
```

```
0000  01 00 5e 00 00 fb 0a 00  27 00 00 0a 08 00 45 00   ··^····· '·····E·
0010  00 38 ca ad 00 00 01 11  15 63 c0 a8 38 01 e0 00   ·8······ ·c··8···
0020  00 fb 14 e9 14 e9 00 24  e7 7c 00 00 00 00 00 01   ·······$ ·|······
0030  00 00 00 00 00 00 04 77  70 61 64 05 6c 6f 63 61   ·······w pad·loca
0040  6c 00 00 01 00 01                                  l·····
```

```
Internet Protocol Version 4 (ip), 20 bytes    Packets: 141 · Displayed: 141 (100.0%)    Profile: Default
```

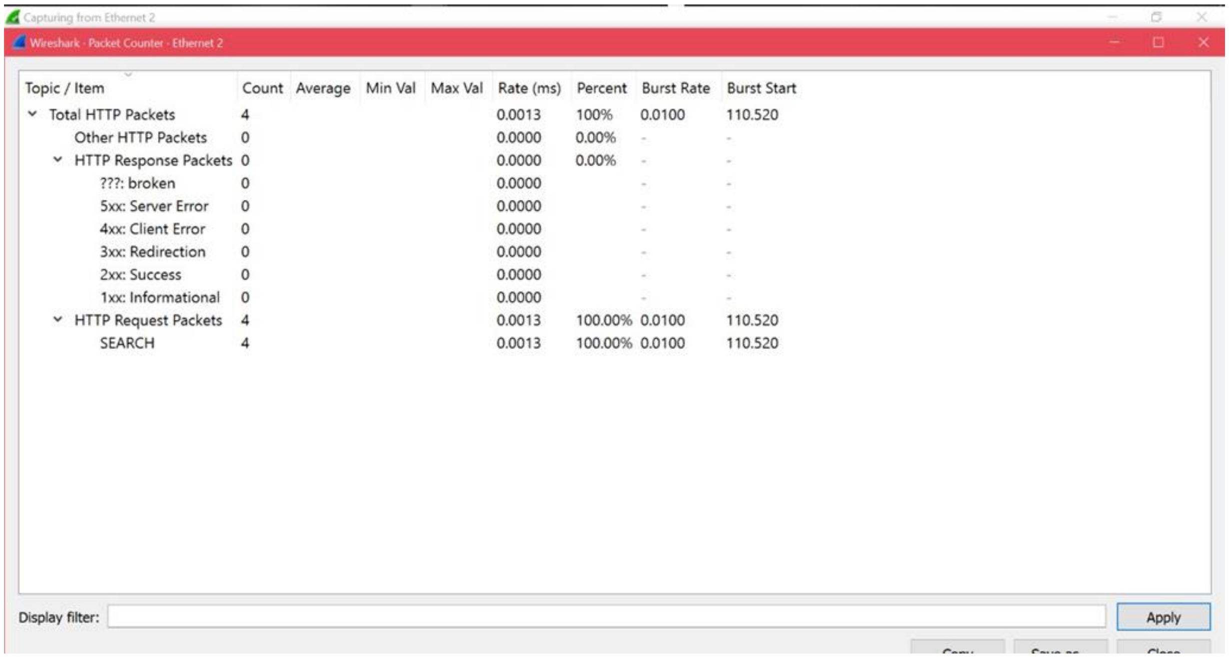
DATA CAPTURING AND ANALYZING THROUGH WIRESHARK



Topic / Item	Count	Average	Min Val	Max Val	Rate (ms)	Percent	Burst Rate	Burst Start
HTTP Requests by HTTP Host	4				0.0013	100%	0.0100	110.520
239.255.255.250:1900	4				0.0013	100.00%	0.0100	110.520
*	4				0.0013	100.00%	0.0100	110.520

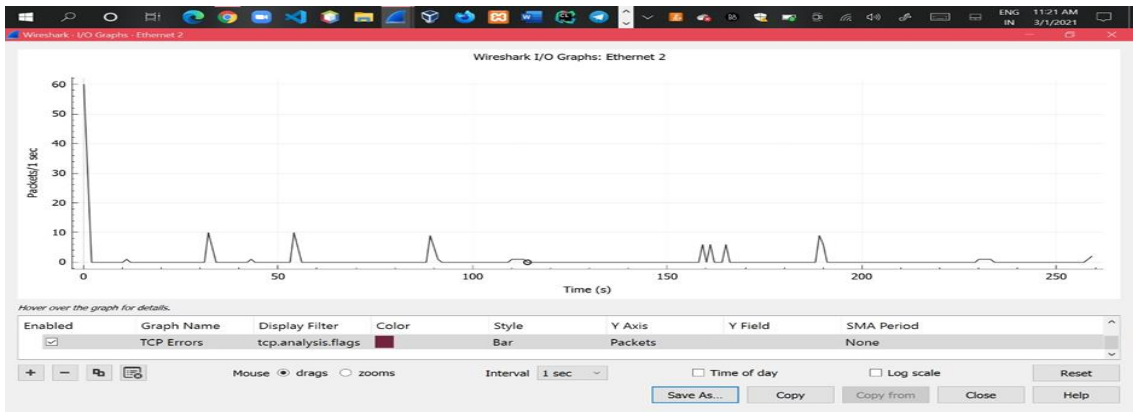
Internet Protocol Version 4 (ip), 20 bytes | Packets: 159 · Displayed: 159 (100.0%) | Profile: Default

DATA REQUEST CAPTURING AND ANALYZING FOR ELIGIBILITY



Topic / Item	Count	Average	Min Val	Max Val	Rate (ms)	Percent	Burst Rate	Burst Start
Total HTTP Packets	4				0.0013	100%	0.0100	110.520
Other HTTP Packets	0				0.0000	0.00%	-	-
HTTP Response Packets	0				0.0000	0.00%	-	-
???: broken	0				0.0000		-	-
5xx: Server Error	0				0.0000		-	-
4xx: Client Error	0				0.0000		-	-
3xx: Redirection	0				0.0000		-	-
2xx: Success	0				0.0000		-	-
1xx: Informational	0				0.0000		-	-
HTTP Request Packets	4				0.0013	100.00%	0.0100	110.520
SEARCH	4				0.0013	100.00%	0.0100	110.520

PACKETS COUNT


Wireshark · I/O Graphs · Ethernet 2
Wireshark I/O Graphs: Ethernet 2
Packets/1 sec
Time (s)
Hover over the graph for details.

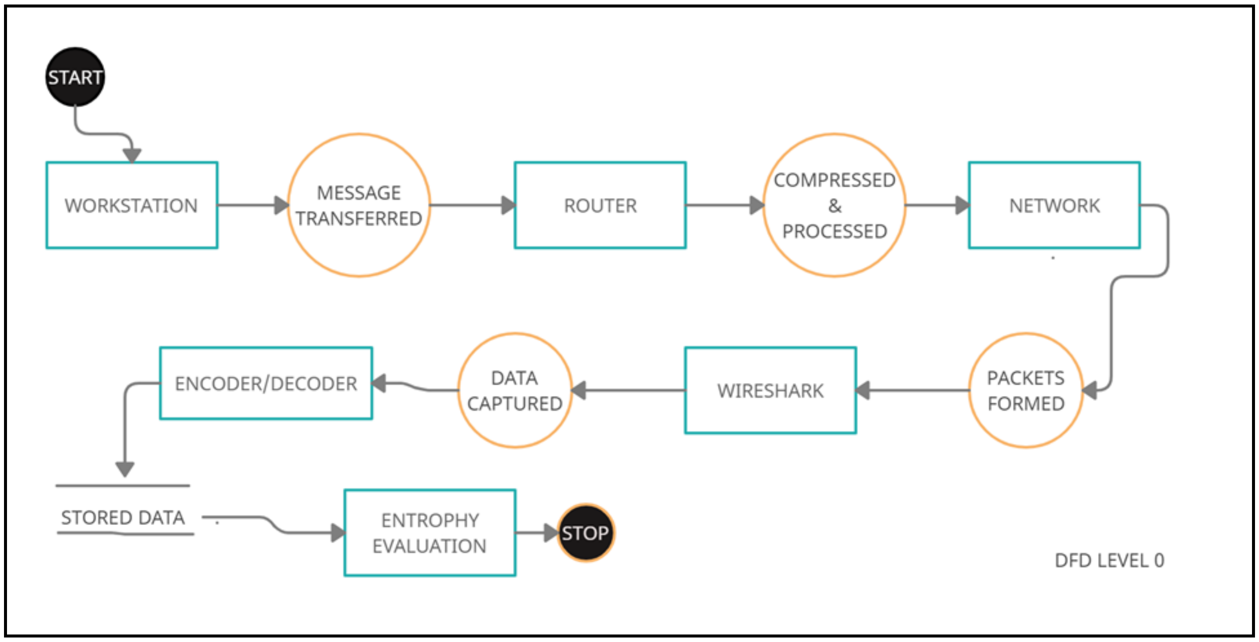
Enabled	Graph Name	Display Filter	Color	Style	Y Axis	Y Field	SMA Period
✓	TCP Errors	tcp.analysis.flags	🟥	Bar	Packets		None

Mouse [✓ drags] zooms Interval 1 sec ☐ Time of day ☐ Log scale

Reset
Save As...
Copy
Copy from
Close
Help

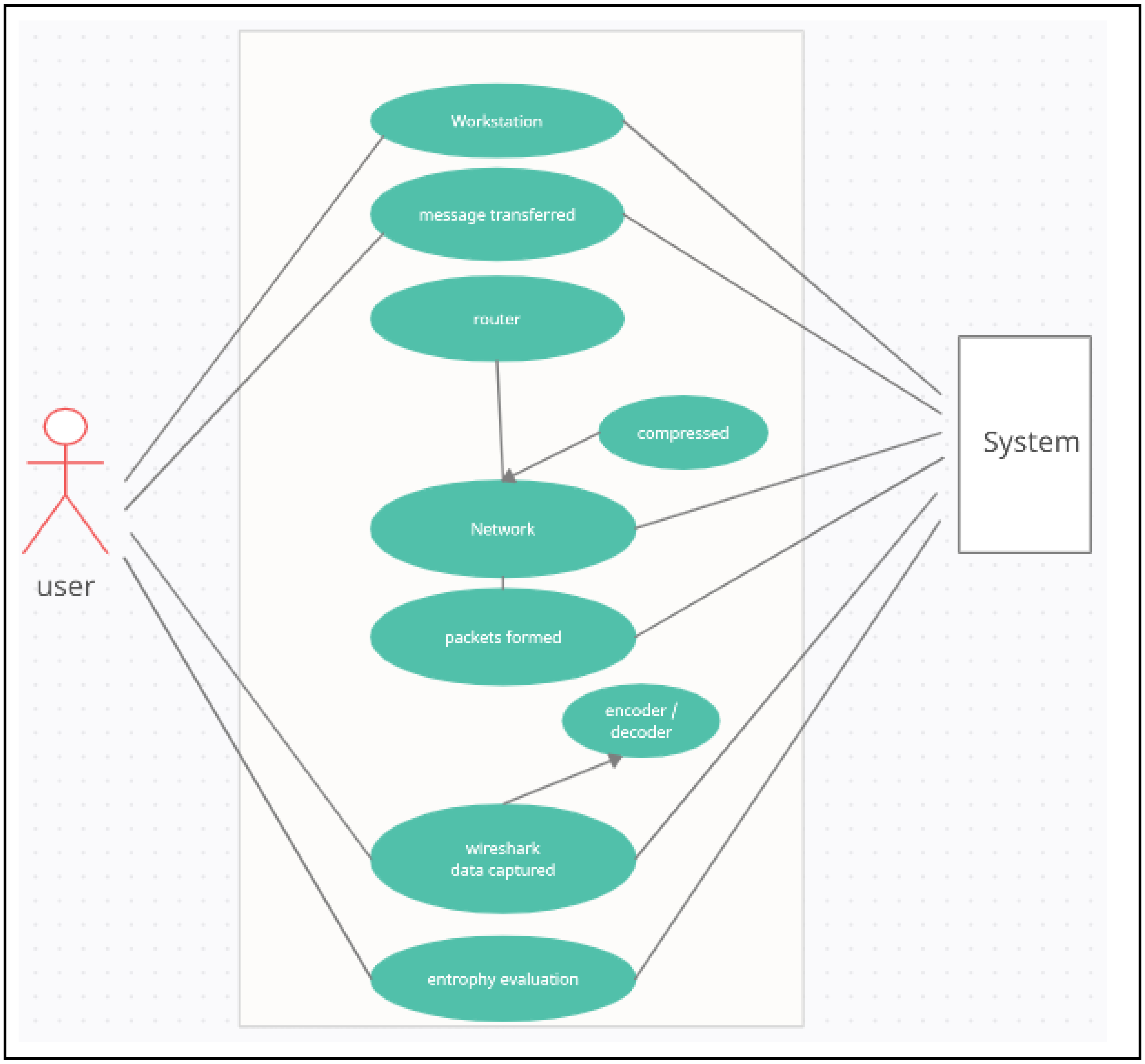
I/O GRAPH AFTER CAPTURING DATA

3.3 DATA FLOW DESIGN:

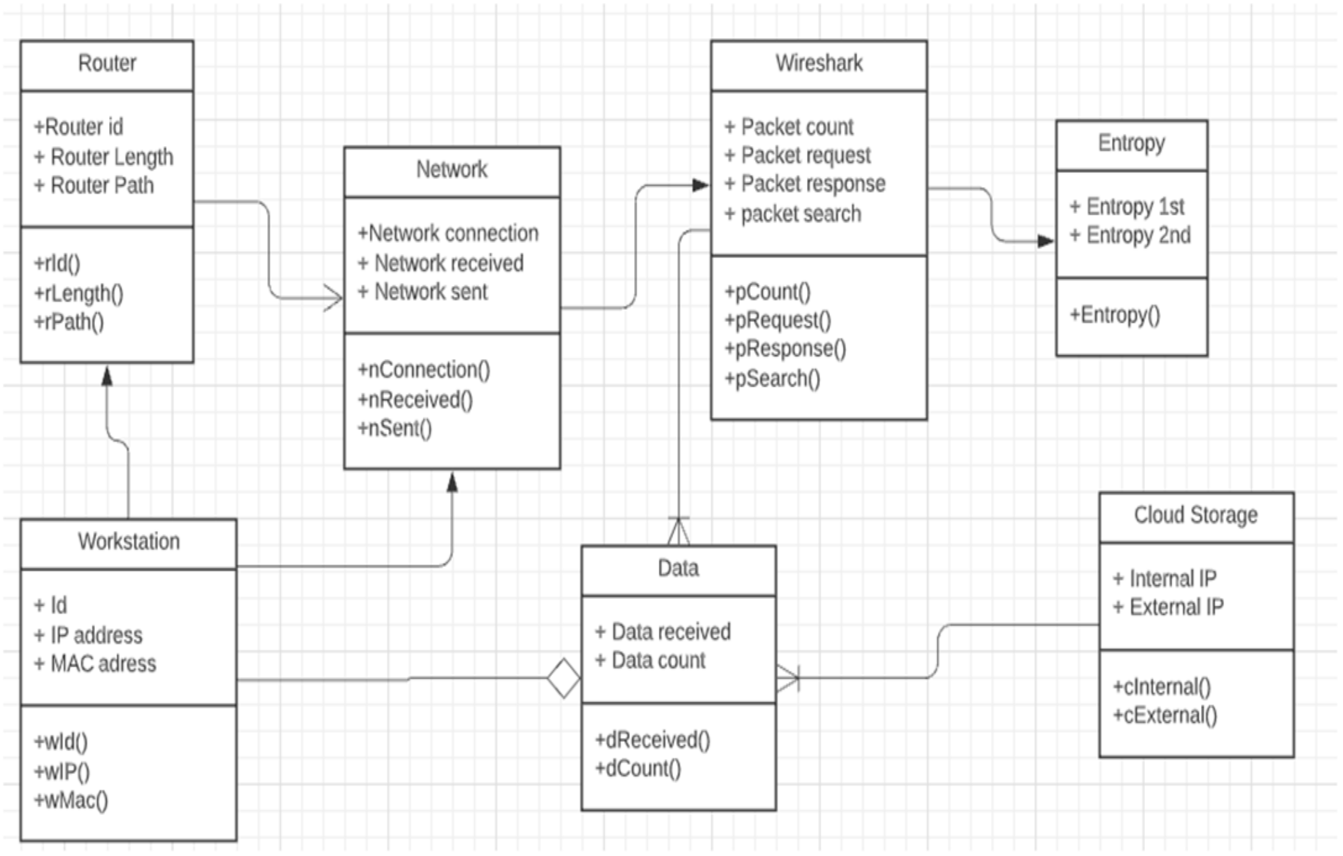


▭	Entity
◯	Process
→	Data flow
═	File/db

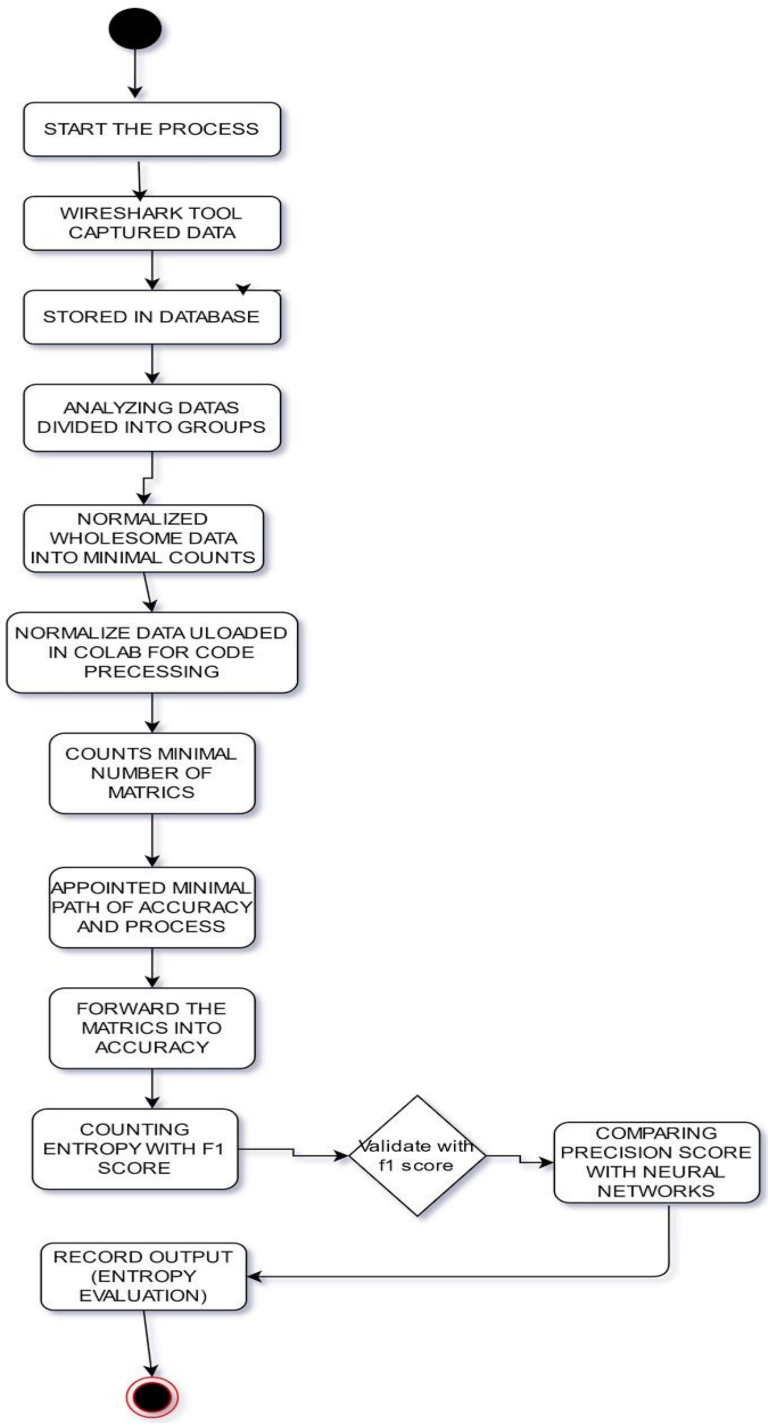
3.3. USECASE DIAGRAM :



3.6 CLASS DIAGRAM:



3.8 STATE-CHART ACTIVITY DIAGRAM


START THE PROCESS
WIRESHARK TOOL CAPTURED DATA
STORED IN DATABASE
ANALYZING DATAS DIVIDED INTO GROUPS
NORMALIZED WHOLESOME DATA INTO MINIMAL COUNTS
NORMALIZE DATA ULOADED IN COLAB FOR CODE PRECESSING
COUNTS MINIMAL NUMBER OF MATRICS
APPOINTED MINIMAL PATH OF ACCURACY AND PROCESS
FORWARD THE MATRICS INTO ACCURACY
COUNTING ENTROPY WITH F1 SCORE
Validate with f1 score
COMPARING PRECISION SCORE WITH NEURAL NETWORKS
RECORD OUTPUT (ENTROPY EVALUATION)

4. IMPLEMENTATION:

The execution of the program is an essential stage of the project where the abstract architecture is compatible with the functional framework. The key stages of deployment are as follows:

- Planning
- Training
- System Testing and
- Transition Preparation

Planning is the first step of implementing the program. Planning is a recommendation on the process and time period to be followed. At the time of deployment, all program workers from various agencies and system development shall be involved. They are verified by the realistic question of monitoring the various behaviours of individuals in their own data processing divisions. Line managers regulated by the implementation coordination committee. The Committee shall consider the suggestions, issues and concerns of the customer community and shall also consider:

- Presence of the program environment;
- Self-selection and assigning of tasks for implementation;
- Consultation with unions and available resources;
- Standby facilities and communication channels;

Methods:

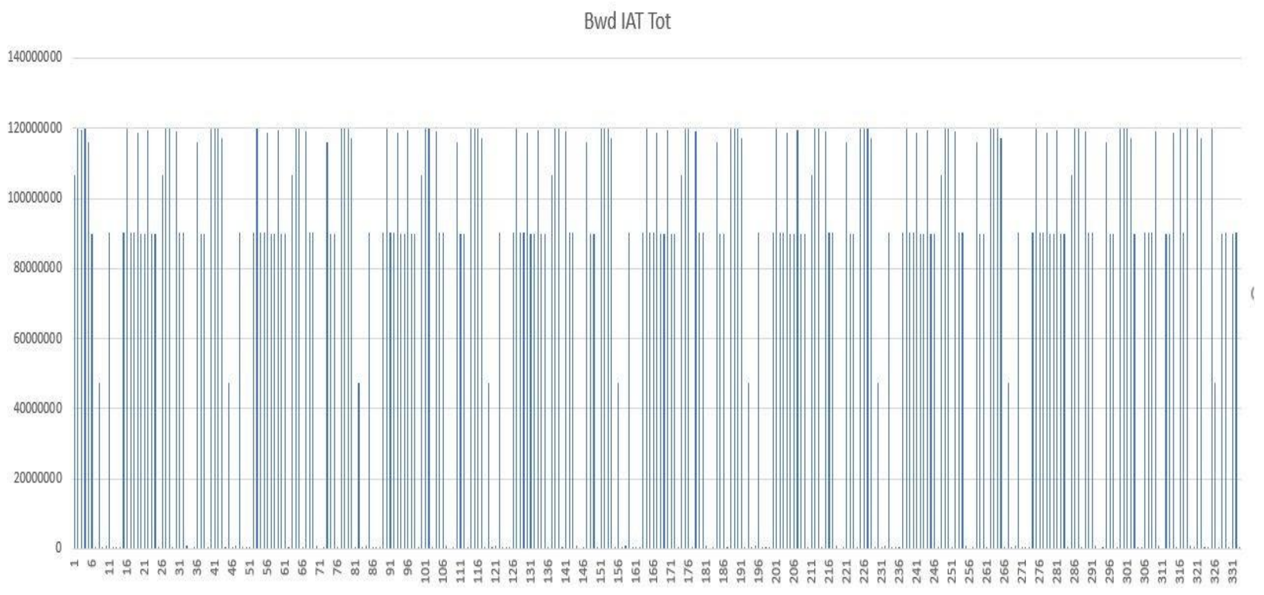
After normalization we used entropy evaluation for encoded and non-encoded traffic classification.

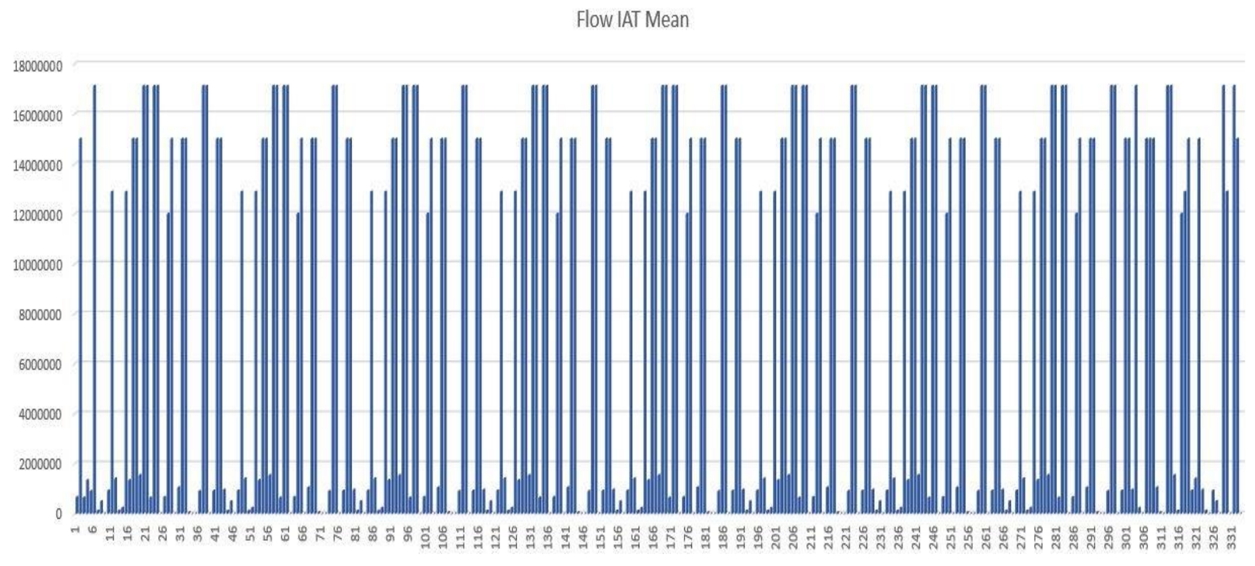
Classification:

The proposed traffic classifier uses a two-stage technique where the coding-based classifier reduces the lower positive rate of the entropy estimate (precision / F1 score) (RNN) based classifier.

Methods for actions classification:

We analysed the relationship between the outcome measure and 23 outstanding statisticians. The figure below shows the density distribution of the "Duration" and "General layout" functions with the duration type (VPN/ NON-VPN). There is no "duration" in the NPV burst, and there is no "general command" in the burst. VPN. The difference between the two types of distributions is obvious, which shows that these classification methods can be used.

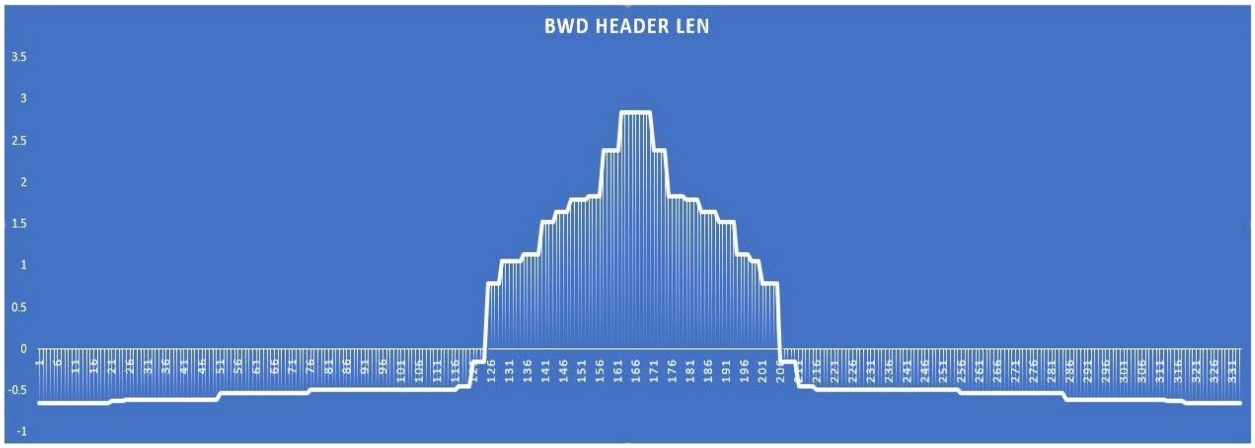


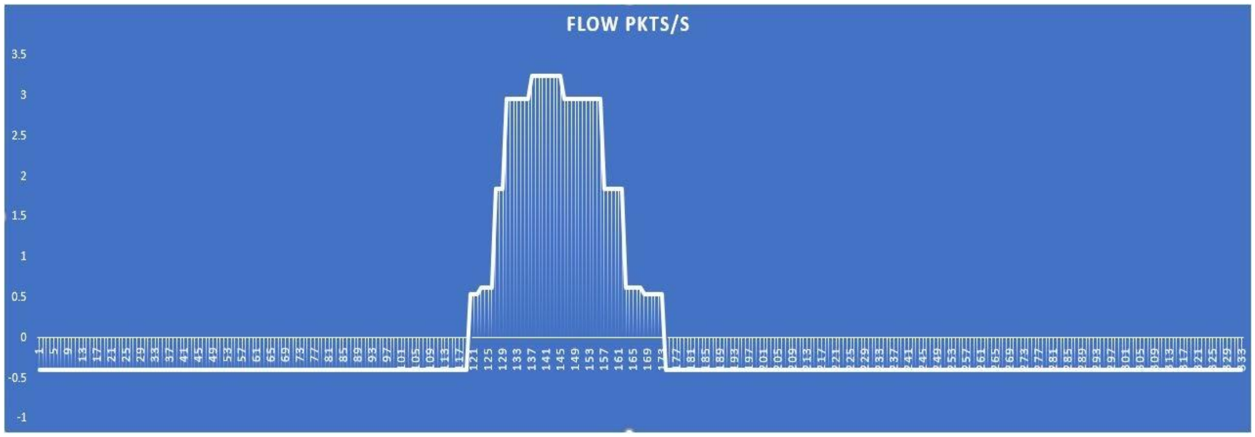


To pre-process this type of dataset, we first hired predominant element analysis (PCA). This methodology significantly reduces the size while keeping the variation of the information, resulting in smaller enter layer sizes. There are 23 features in the dataset. After PCA processing, ten major additives with a variance of more than 20% were calculated. To keep the information variance at 0.20, we chose the top ten most important ingredients.

After that we have remaining dataset put into RNN set of procedure where we have implemented

We gauge that the payload for basic literary substance messages will be encoded in ASCII or ANSI, with printable character esteems going from 32 to 127. Since literary substance messages may seem arbitrary when utilizing an entropy-based technique, we made a bunch of decisions that recognize enormous text based substance blocks inside the payload. The likelihood of a person from an irregular source being inside the scope of 32 to 127 is roughly 20%. The payload is undoubtedly decoded, particularly if an enormous bit of the characters in the start of the bundle are on this kind. As an outcome, we incorporated a watch that says if the negligible portion of bytes with values somewhere in the range of 32 and 127 is more noteworthy than 75%, the stream is decoded. We presently don't look at the bundle's entire payload since we need to decrease handling time. Just the initial 96 bytes are assessed for the coding-based total classifier.





The following are the detailed steps:

Step 1:

As indicated by the five-tuple, network traffic is caught. A connection exists between three continuous SYN parcels and the last FIN or RST bundle in TCP transmission. A connection is likewise decided for UDP traffic dependent on the period between the main parcel got and no bundle for 60 seconds.

The main parcel of a solitary connection is extricated, and it is resolved if the connection is dynamic.

The underlying bundle has a length of in excess of 1024 bytes. On the off chance that it is, its payload will be separated; if not, it will be erased and the following bundle will be removed until all N parcels have been extricated.The Shannon entropy equation is utilized to register the entropy worth of each character in the whole information bundle utilizing the equivalent dividing strategy.

Step 3:

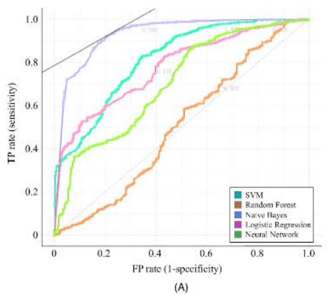
Every one of the N characters in the payload of the removed parcel is used as a bunch of Monte Carlo reenactment focuses. The x-point of the arrange hub is dictated by the main N/2 characters, while the y-point is controlled by the keep going N/2 characters. The assessed esteem is determined dependent on the quantity of arranged focuses that fall into the circle, just as the blunder with a genuine worth.

To get characterization results, the entropy esteem H and Monte Carol assessment esteem mistake P are normalized, and the two highlights are then taken care of into the C4.5 choice tree classifier.

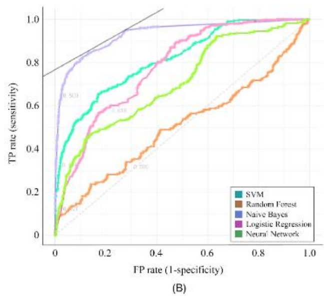
ENCODED TRAFFIC CLASSIFICATION:

We contrasted customary framework investigation philosophies with our mixed methodology for sorting utilizing the equivalent dataset

.

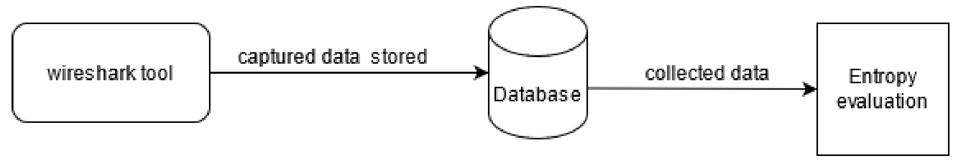


(A)

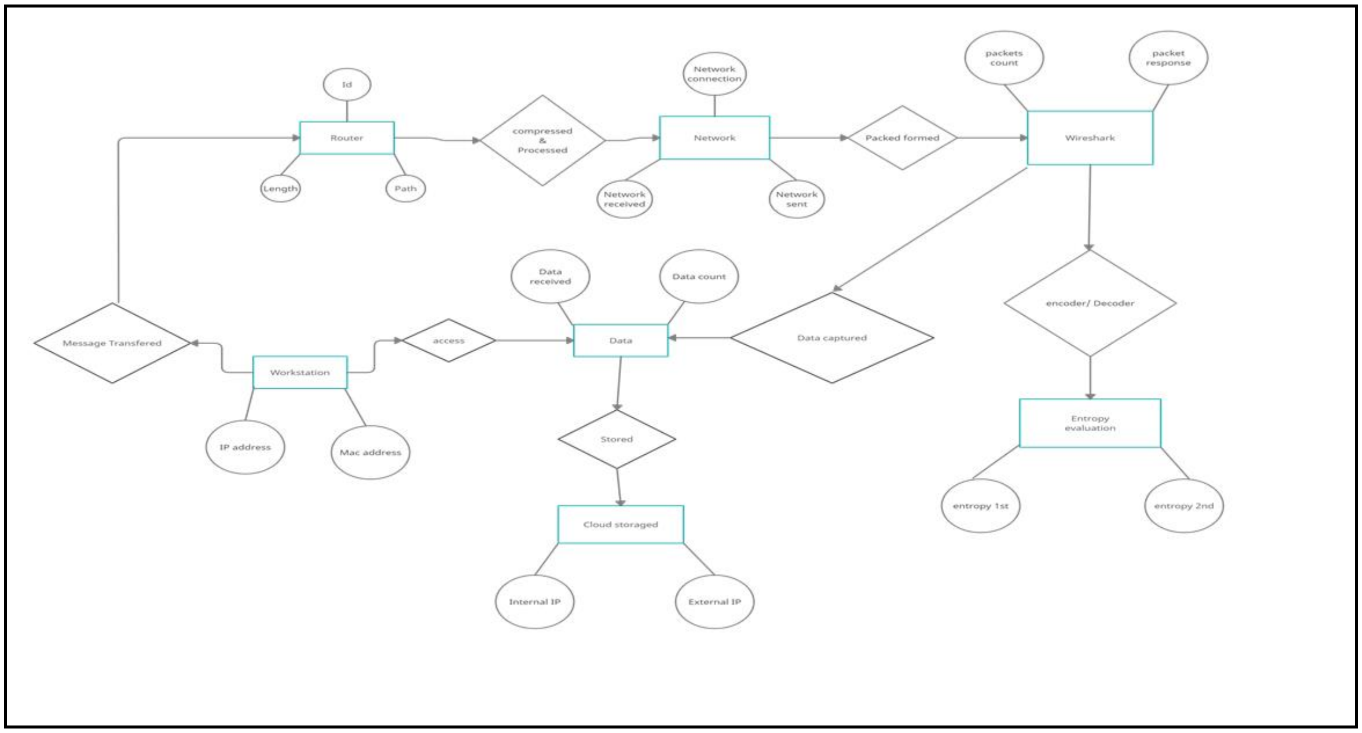


(B)

4.1. DATABASE DESIGN:

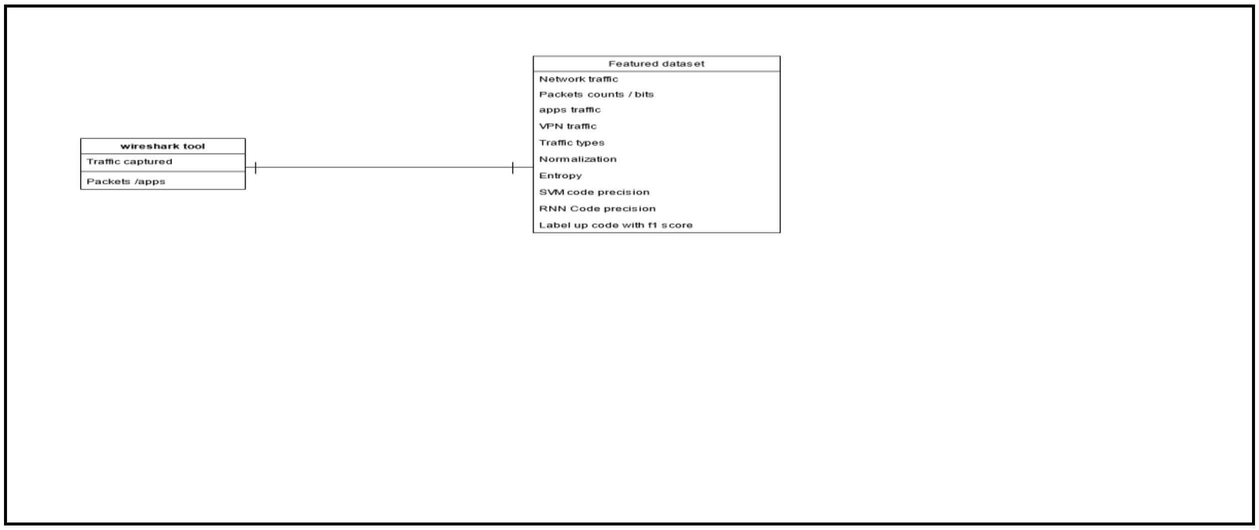


4.1.1. **ER DIAGRAM**

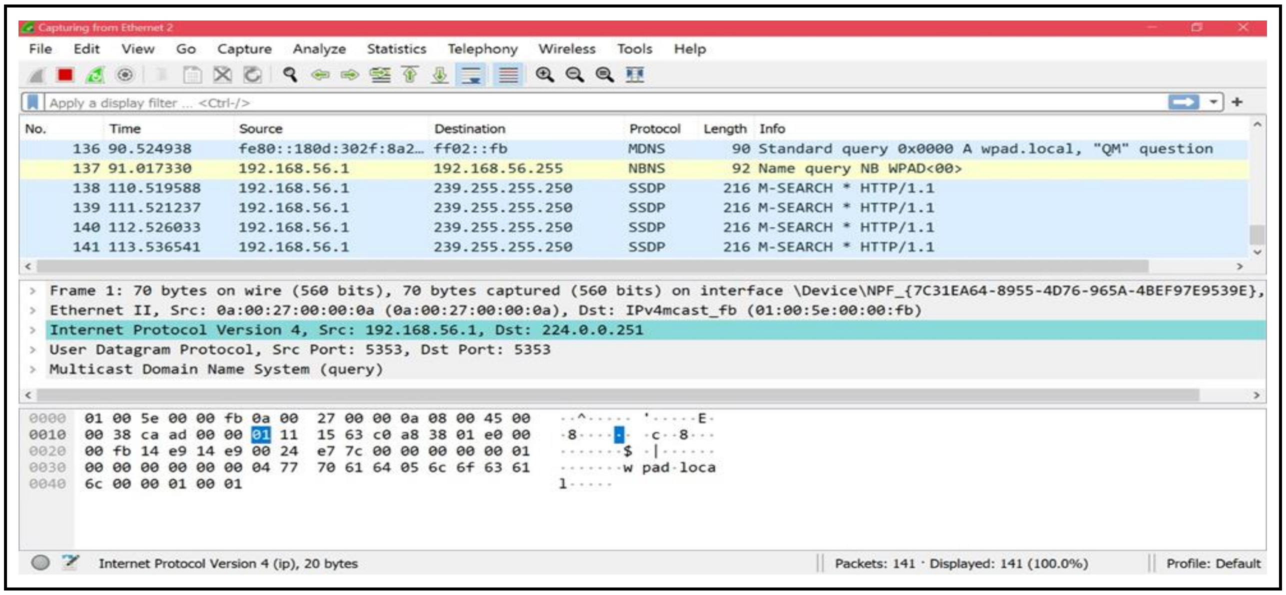


▭ Entity
○ Attributes
◇ process

4.1.2. RELATIONAL MODEL:



4.2. USER INTERFACE:


Capturing from Ethernet 2
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

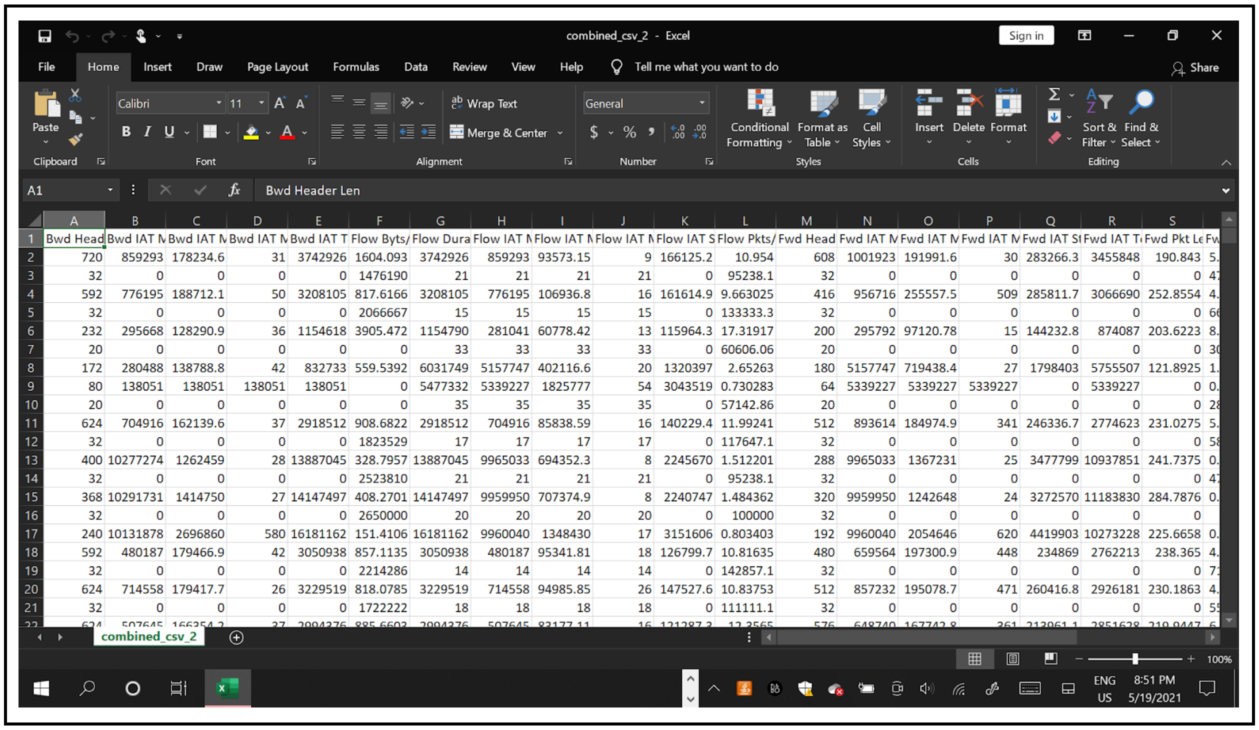
```
No.   Time         Source                 Destination        Protocol  Length Info
  136 90.524938    fe80::180d:302f:8a2…   ff02::fb           MDNS          90 Standard query 0x0000 A wpad.local, "QM" question
  137 91.017330    192.168.56.1           192.168.56.255     NBNS          92 Name query NB WPAD<00>
  138 110.519588   192.168.56.1           239.255.255.250    SSDP         216 M-SEARCH * HTTP/1.1
  139 111.521237   192.168.56.1           239.255.255.250    SSDP         216 M-SEARCH * HTTP/1.1
  140 112.526033   192.168.56.1           239.255.255.250    SSDP         216 M-SEARCH * HTTP/1.1
  141 113.536541   192.168.56.1           239.255.255.250    SSDP         216 M-SEARCH * HTTP/1.1

> Frame 1: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface \Device\NPF_{7C31EA64-8955-4D76-965A-4BEF97E9539E},
> Ethernet II, Src: 0a:00:27:00:00:0a (0a:00:27:00:00:0a), Dst: IPv4mcast_fb (01:00:5e:00:00:fb)
> Internet Protocol Version 4, Src: 192.168.56.1, Dst: 224.0.0.251
> User Datagram Protocol, Src Port: 5353, Dst Port: 5353
> Multicast Domain Name System (query)

0000  01 00 5e 00 00 fb 0a 00  27 00 00 0a 08 00 45 00   ··^·····  '·····E·
0010  00 38 ca ad 00 00 01 11  15 63 c0 a8 38 01 e0 00   ·8······  ·c··8···
0020  00 fb 14 e9 14 e9 00 24  e7 7c 00 00 00 00 00 01   ·······$  ·|······
0030  00 00 00 00 00 00 04 77  70 61 64 05 6c 6f 63 61   ·······w  pad·loca
0040  6c 00 00 01 00 01                                  l·····

Internet Protocol Version 4 (ip), 20 bytes        Packets: 141 · Displayed: 141 (100.0%)        Profile: Default
```

4.3. MIDDLEWARE:

- **Google Colab**: The purpose of Google Colab is to make package management and execution easier. It's a web-based utility that's both at large and non-proprietary.
- **Python with NumPy**: NumPy is a primary science programming system for Python. Among other stuff
- **Python with Pandas**: Pandas is an open-source data analysis and manipulation tool that is fast, effective, scalable, and simple to use.

-

4.4. DATASETS:



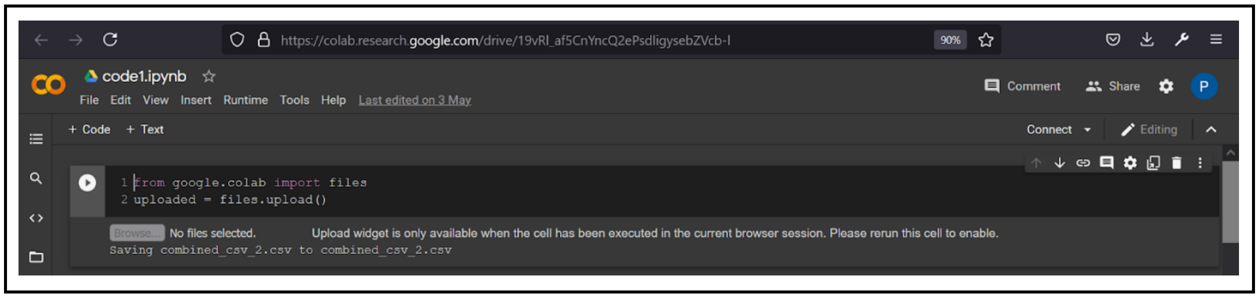
combined_csv_2 - Excel

A1 Bwd Header Len

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	
1	Bwd Head	Bwd IAT M	Bwd IAT M	Bwd IAT M	Bwd IAT T	Flow Byts/	Flow Dura	Flow IAT M	Flow IAT M	Flow IAT M	Flow IAT S	Flow Pkts/	Fwd Head	Fwd IAT M	Fwd IAT M	Fwd IAT M	Fwd IAT St	Fwd IAT T	Fwd Pkt Le	Fw
2	720	859293	178234.6	31	3742926	1604.093	3742926	859293	93573.15	9	166125.2	10.954	608	1001923	191991.6	30	283266.3	3455848	190.843	5.
3	32	0	0	0	0	1476190	21	21	21	21	0	95238.1	32	0	0	0	0	0	0	47
4	592	776195	188712.1	50	3208105	817.6166	3208105	776195	106936.8	16	161614.9	9.663025	416	956716	255557.5	509	285811.7	3066690	252.8554	4.
5	32	0	0	0	0	2066667	15	15	15	15	0	133333.3	32	0	0	0	0	0	0	66
6	232	295668	128290.9	36	1154618	3905.472	1154790	281041	60778.42	13	115964.3	17.31917	200	295792	97120.78	15	144232.8	874087	203.6223	8.
7	20	0	0	0	0	0	33	33	33	33	0	60606.06	20	0	0	0	0	0	0	30
8	172	280488	138788.8	42	832733	559.5392	6031749	5157747	402116.6	20	1320397	2.65263	180	5157747	719438.4	27	1798403	5755507	121.8925	1.
9	80	138051	138051	138051	138051	0	5477332	5339227	1825777	54	3043519	0.730283	64	5339227	5339227	5339227	0	5339227	0	0.
10	20	0	0	0	0	0	35	35	35	35	0	57142.86	20	0	0	0	0	0	0	28
11	624	704916	162139.6	37	2918512	908.6822	2918512	704916	85838.59	16	140229.4	11.99241	512	893614	184974.9	341	246336.7	2774623	231.0275	5.
12	32	0	0	0	0	1823529	17	17	17	17	0	117647.1	32	0	0	0	0	0	0	58
13	400	10277274	1262459	28	13887045	328.7957	13887045	9965033	694352.3	8	2245670	1.512201	288	9965033	1367231	25	3477799	10937851	241.7375	0.
14	32	0	0	0	0	2523810	21	21	21	21	0	95238.1	32	0	0	0	0	0	0	47
15	368	10291731	1414750	27	14147497	408.2701	14147497	9959950	707374.9	8	2240747	1.484362	320	9959950	1242648	24	3272570	11183830	284.7876	0.
16	32	0	0	0	0	2650000	20	20	20	20	0	100000	32	0	0	0	0	0	0	
17	240	10131878	2696860	580	16181162	151.4106	16181162	9960040	1348430	17	3151606	0.803403	192	9960040	2054646	620	4419903	10273228	225.6658	0.
18	592	480187	179466.9	42	3050938	857.1135	3050938	480187	95341.81	18	126799.7	10.81635	480	659564	197300.9	448	234869	2762213	238.365	4.
19	32	0	0	0	0	2214286	14	14	14	14	0	142857.1	32	0	0	0	0	0	0	71
20	624	714558	179417.7	26	3229519	818.0785	3229519	714558	94985.85	26	147527.6	10.83753	512	857232	195078.7	471	260416.8	2926181	230.1863	4.
21	32	0	0	0	0	1722222	18	18	18	18	0	111111.1	32	0	0	0	0	0	0	55

combined_csv_2

4.5. CODE



https://colab.research.google.com/drive/19vRl_af5CnYncQ2ePsdligysebZVcb-l

code1.ipynb

```
from google.colab import files
uploaded = files.upload()
```

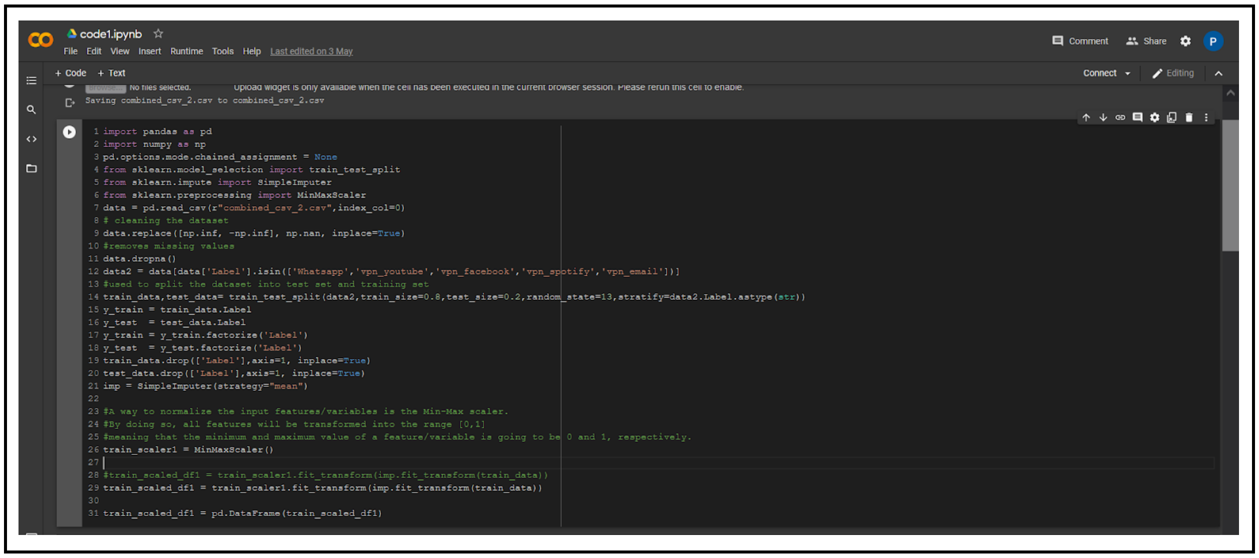
Browse... No files selected. Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving combined_csv_2.csv to combined_csv_2.csv
```

```
import pandas as pd
import numpy as np
pd.options.mode.chained_assignment = None
from sklearn.model_selection import train_test_split
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import MinMaxScaler
data = pd.read_csv(r"combined_csv_2.csv",index_col=0)
# cleaning the dataset
data.replace([np.inf, -np.inf], np.nan, inplace=True)
#removes missing values
data.dropna()
data2 = data[data['Label'].isin(['Whatsapp','vpn_youtube',
                        'vpn_facebook','vpn_spotify','vpn_email'])]
#used to split the dataset into test set and training set
train_data,test_data= train_test_split(data2,train_size=0.8,test_size=0.2,
                        random_state=13,stratify=data2.Label.astype(str))
y_train = train_data.Label
y_test  = test_data.Label
y_train = y_train.factorize('Label')
y_test  = y_test.factorize('Label')
train_data.drop(['Label'],axis=1, inplace=True)
test_data.drop(['Label'],axis=1, inplace=True)
imp = SimpleImputer(strategy="mean")

#A way to normalize the input features/variables is the Min-Max scaler.
#By doing so, all features will be transformed into the range [0,1]
#meaning that the minimum and maximum value of a feature/variable
                                is going to be 0 and 1, respectively.
train_scaler1 = MinMaxScaler()

#train_scaled_df1 = train_scaler1.fit_transform(imp.fit_transform(train_data))
train_scaled_df1 = train_scaler1.fit_transform(imp.fit_transform(train_data))

train_scaled_df1 = pd.DataFrame(train_scaled_df1)
```

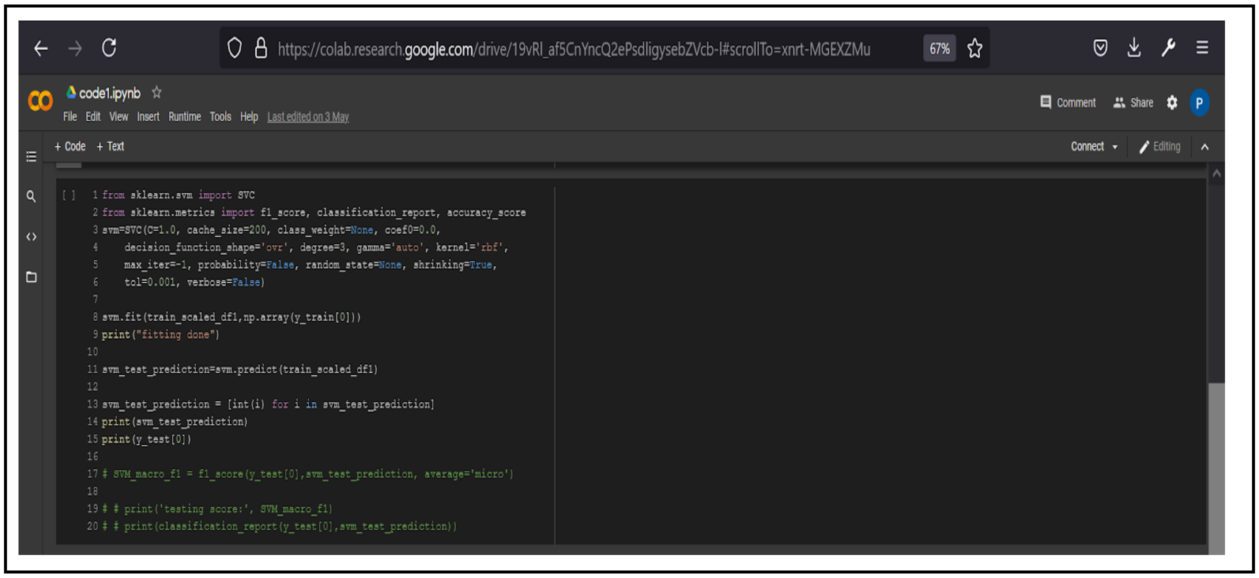
SVM Code

```
from sklearn.svm import SVC
from sklearn.metrics import f1_score, classification_report, accuracy_score
svm=SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
        decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf',
        max_iter=-1, probability=False, random_state=None, shrinking=True,
        tol=0.001, verbose=False)

svm.fit(train_scaled_df1,np.array(y_train[0]))
print("fitting done")

svm_test_prediction=svm.predict(train_scaled_df1)

svm_test_prediction = [int(i) for i in svm_test_prediction]
print(svm_test_prediction)
print(y_test[0])

# SVM_macro_f1 = f1_score(y_test[0],svm_test_prediction, average='micro')

# # print('testing score:', SVM_macro_f1)
# # print(classification_report(y_test[0],svm_test_prediction))
```


code1.ipynb
File Edit View Insert Runtime Tools Help Last edited on 3 May
Comment Share
+ Code + Text
Connect Editing

```
No files selected.    Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving combined_csv_2.csv to combined_csv_2.csv
```

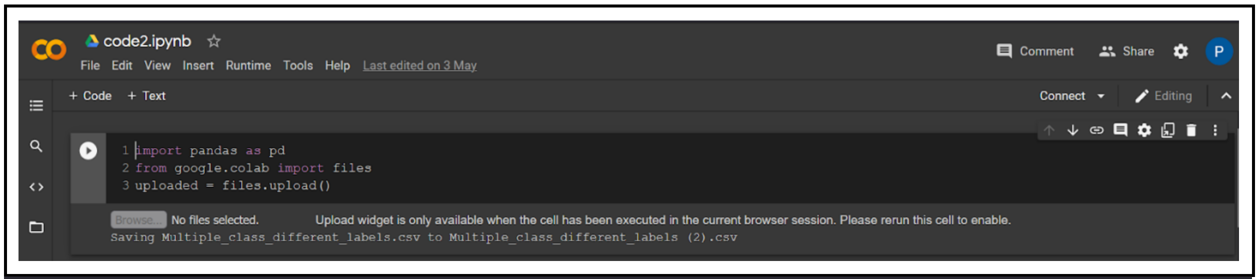
```
import pandas as pd
import numpy as np
pd.options.mode.chained_assignment = None
from sklearn.model_selection import train_test_split
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import MinMaxScaler
data = pd.read_csv(r"combined_csv_2.csv",index_col=0)
# cleaning the dataset
data.replace([np.inf, -np.inf], np.nan, inplace=True)
#removes missing values
data.dropna()
data2 = data[data['Label'].isin(['Whatsapp','vpn_youtube','vpn_facebook','vpn_spotify','vpn_email'])]
#used to split the dataset into test set and training set
train_data,test_data= train_test_split(data2,train_size=0.8,test_size=0.2,random_state=13,stratify=data2.Label.astype(str))
y_train = train_data.Label
y_test  = test_data.Label
y_train = y_train.factorize('Label')
y_test  = y_test.factorize('Label')
train_data.drop(['Label'],axis=1, inplace=True)
test_data.drop(['Label'],axis=1, inplace=True)
imp = SimpleImputer(strategy="mean")

#A way to normalize the input features/variables is the Min-Max scaler.
#By doing so, all features will be transformed into the range [0,1]
#meaning that the minimum and maximum value of a feature/variable is going to be 0 and 1, respectively.
train_scaler1 = MinMaxScaler()

#train_scaled_df1 = train_scaler1.fit_transform(imp.fit_transform(train_data))
train_scaled_df1 = train_scaler1.fit_transform(imp.fit_transform(train_data))

train_scaled_df1 = pd.DataFrame(train_scaled_df1)
```



https://colab.research.google.com/drive/19vRl_af5CnYncQ2ePsdligysebZVcb-l#scrollTo=xnrt-MGEXZMu
67%
code1.ipynb
File Edit View Insert Runtime Tools Help Last edited on 3 May
Comment Share
+ Code + Text
Connect Editing

```
from sklearn.svm import SVC
from sklearn.metrics import f1_score, classification_report, accuracy_score
svm=SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)

svm.fit(train_scaled_df1,np.array(y_train[0]))
print("fitting done")

svm_test_prediction=svm.predict(train_scaled_df1)

svm_test_prediction = [int(i) for i in svm_test_prediction]
print(svm_test_prediction)
print(y_test[0])

# SVM_macro_f1 = f1_score(y_test[0],svm_test_prediction, average='micro')

# # print('testing score:', SVM_macro_f1)
# # print(classification_report(y_test[0],svm_test_prediction))
```

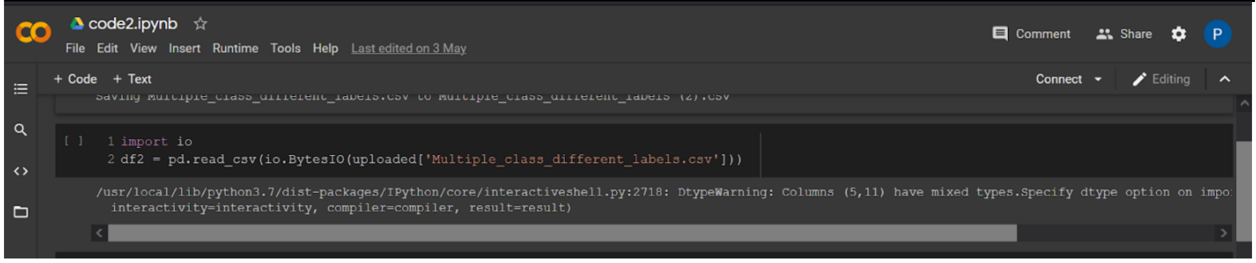
RNN CODE:



code2.ipynb
File Edit View Insert Runtime Tools Help Last edited on 3 May
Comment Share
+ Code + Text
Connect Editing
Browse...

```
import pandas as pd
from google.colab import files
uploaded = files.upload()
```

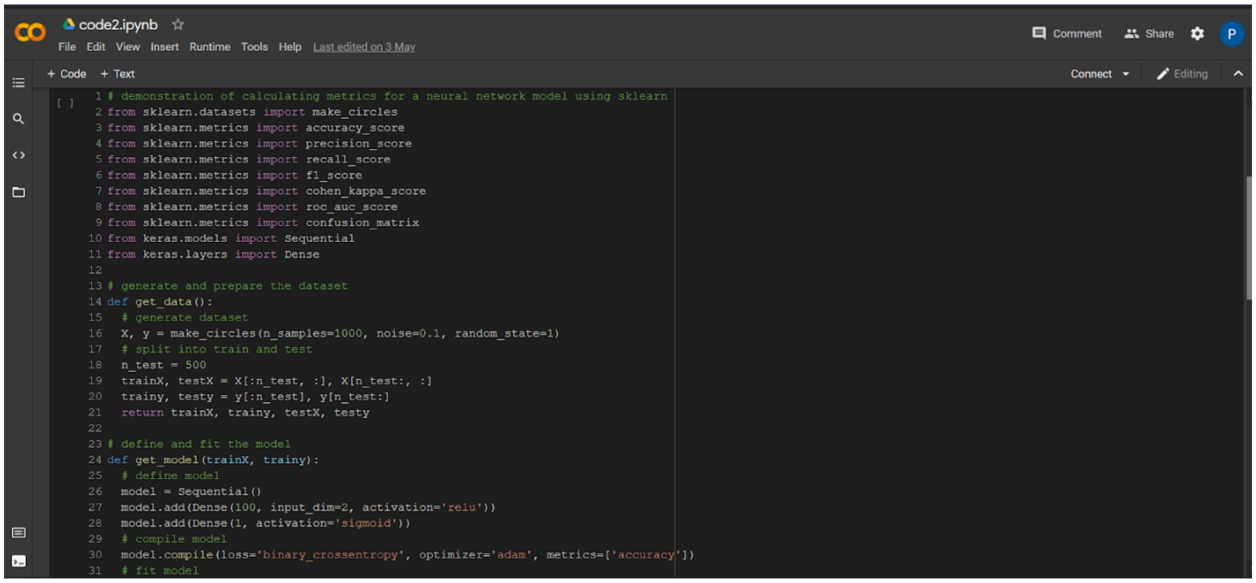
```
No files selected.    Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving Multiple_class_different_labels.csv to Multiple_class_different_labels (2).csv
```



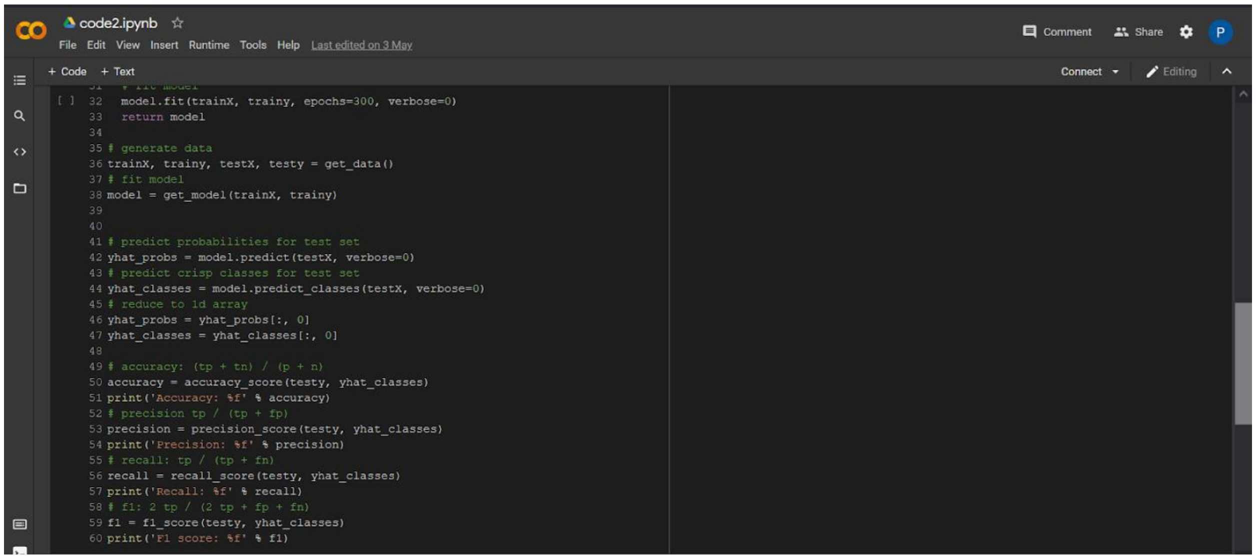
code2.ipynb
File Edit View Insert Runtime Tools Help Last edited on 3 May
Comment Share
+ Code + Text
Connect Editing

```
import io
df2 = pd.read_csv(io.BytesIO(uploaded['Multiple_class_different_labels.csv']))
```

```
/usr/local/lib/python3.7/dist-packages/IPython/core/interactiveshell.py:2718: DtypeWarning: Columns (5,11) have mixed types.Specify dtype option on impo
  interactivity=interactivity, compiler=compiler, result=result)
```


code2.ipynb
File Edit View Insert Runtime Tools Help Last edited on 3 May
Comment Share
+ Code + Text
Connect Editing

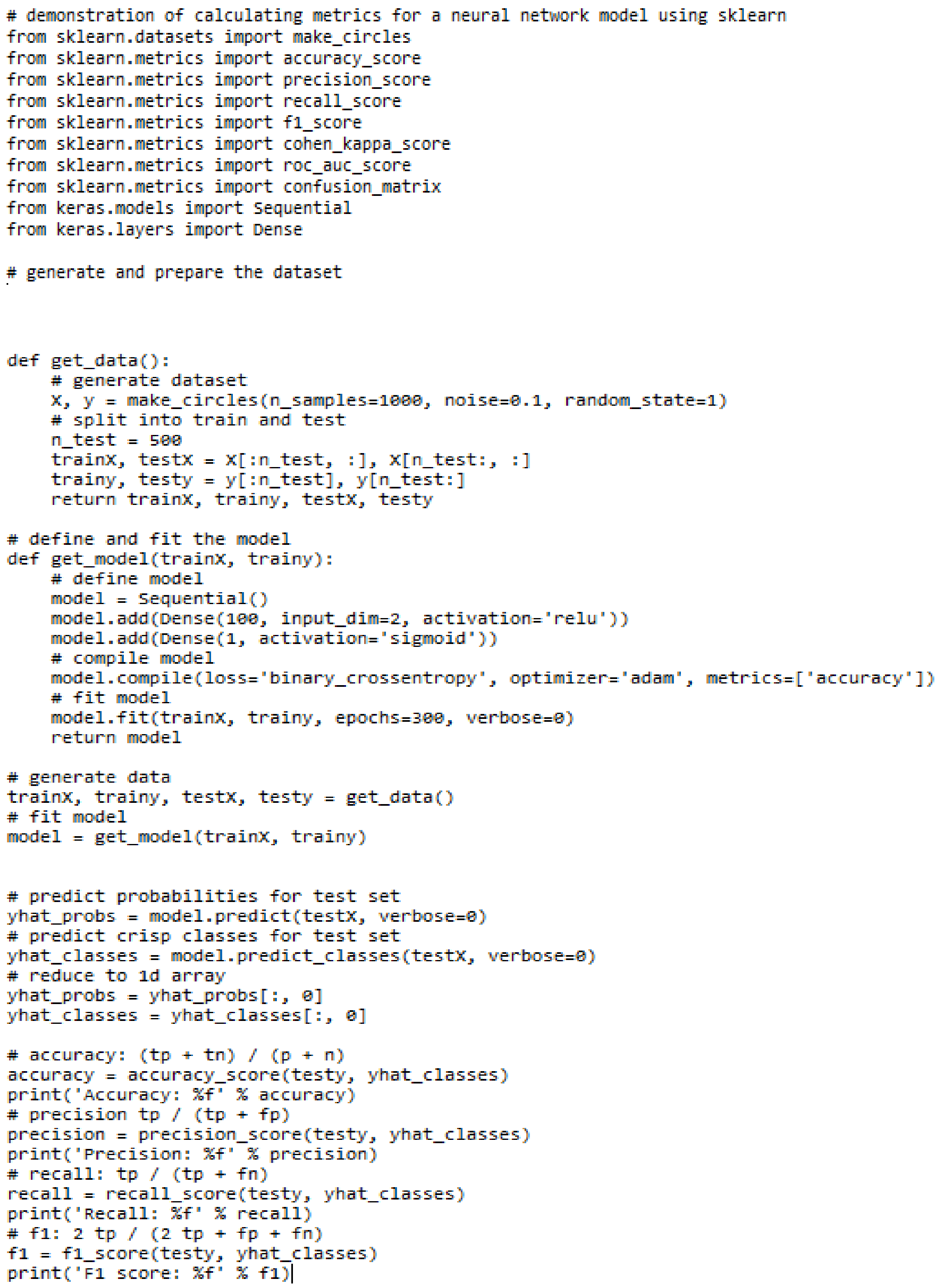
```python
# demonstration of calculating metrics for a neural network model using sklearn
from sklearn.datasets import make_circles
from sklearn.metrics import accuracy_score
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import f1_score
from sklearn.metrics import cohen_kappa_score
from sklearn.metrics import roc_auc_score
from sklearn.metrics import confusion_matrix
from keras.models import Sequential
from keras.layers import Dense

# generate and prepare the dataset
def get_data():
  # generate dataset
  X, y = make_circles(n_samples=1000, noise=0.1, random_state=1)
  # split into train and test
  n_test = 500
  trainX, testX = X[:n_test, :], X[n_test:, :]
  trainy, testy = y[:n_test], y[n_test:]
  return trainX, trainy, testX, testy

# define and fit the model
def get_model(trainX, trainy):
  # define model
  model = Sequential()
  model.add(Dense(100, input_dim=2, activation='relu'))
  model.add(Dense(1, activation='sigmoid'))
  # compile model
  model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
  # fit model
```


code2.ipynb
File Edit View Insert Runtime Tools Help Last edited on 3 May
Comment Share
+ Code + Text
Connect Editing

```python
  model.fit(trainX, trainy, epochs=300, verbose=0)
  return model

# generate data
trainX, trainy, testX, testy = get_data()
# fit model
model = get_model(trainX, trainy)


# predict probabilities for test set
yhat_probs = model.predict(testX, verbose=0)
# predict crisp classes for test set
yhat_classes = model.predict_classes(testX, verbose=0)
# reduce to 1d array
yhat_probs = yhat_probs[:, 0]
yhat_classes = yhat_classes[:, 0]

# accuracy: (tp + tn) / (p + n)
accuracy = accuracy_score(testy, yhat_classes)
print('Accuracy: %f' % accuracy)
# precision tp / (tp + fp)
precision = precision_score(testy, yhat_classes)
print('Precision: %f' % precision)
# recall: tp / (tp + fn)
recall = recall_score(testy, yhat_classes)
print('Recall: %f' % recall)
# f1: 2 tp / (2 tp + fp + fn)
f1 = f1_score(testy, yhat_classes)
print('F1 score: %f' % f1)
```

```python
import pandas as pd
from google.colab import files
uploaded = files.upload()
```

```python
import io
df2 = pd.read_csv(io.BytesIO(uploaded['Multiple_class_different_labels.csv']))
```



```
# demonstration of calculating metrics for a neural network model using sklearn
from sklearn.datasets import make_circles
from sklearn.metrics import accuracy_score
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import f1_score
from sklearn.metrics import cohen_kappa_score
from sklearn.metrics import roc_auc_score
from sklearn.metrics import confusion_matrix
from keras.models import Sequential
from keras.layers import Dense

# generate and prepare the dataset



def get_data():
    # generate dataset
    X, y = make_circles(n_samples=1000, noise=0.1, random_state=1)
    # split into train and test
    n_test = 500
    trainX, testX = X[:n_test, :], X[n_test:, :]
    trainy, testy = y[:n_test], y[n_test:]
    return trainX, trainy, testX, testy

# define and fit the model
def get_model(trainX, trainy):
    # define model
    model = Sequential()
    model.add(Dense(100, input_dim=2, activation='relu'))
    model.add(Dense(1, activation='sigmoid'))
    # compile model
    model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
    # fit model
    model.fit(trainX, trainy, epochs=300, verbose=0)
    return model

# generate data
trainX, trainy, testX, testy = get_data()
# fit model
model = get_model(trainX, trainy)


# predict probabilities for test set
yhat_probs = model.predict(testX, verbose=0)
# predict crisp classes for test set
yhat_classes = model.predict_classes(testX, verbose=0)
# reduce to 1d array
yhat_probs = yhat_probs[:, 0]
yhat_classes = yhat_classes[:, 0]

# accuracy: (tp + tn) / (p + n)
accuracy = accuracy_score(testy, yhat_classes)
print('Accuracy: %f' % accuracy)
# precision tp / (tp + fp)
precision = precision_score(testy, yhat_classes)
print('Precision: %f' % precision)
# recall: tp / (tp + fn)
recall = recall_score(testy, yhat_classes)
print('Recall: %f' % recall)
# f1: 2 tp / (2 tp + fp + fn)
f1 = f1_score(testy, yhat_classes)
print('F1 score: %f' % f1)
```

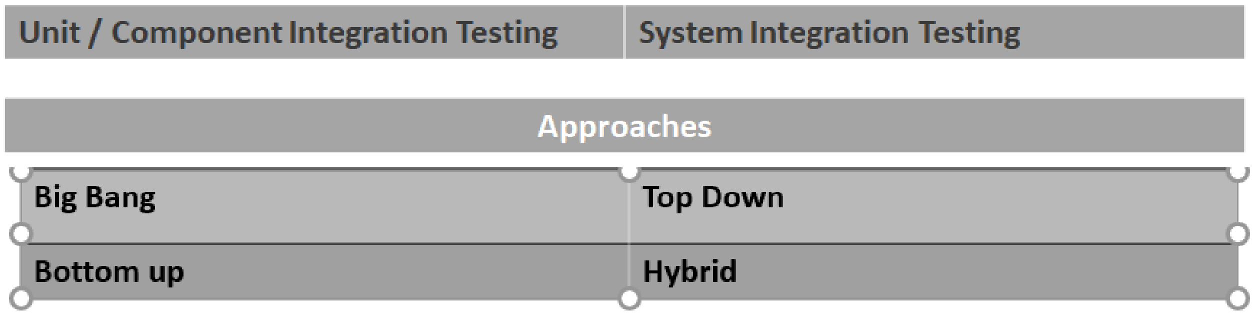
5. VERIFICATION & VALIDATION

5.1. **Unit Testing**

Singular units or segments of programming are tried in this kind of programming testing. The objective is to guarantee that every unit of programming code fills in as expected. Unit testing is done by engineers all through the turn of events (coding) period of an application.

5.2. **Integration Testing**

- Singular units/segments are combined and tried collectively at this degree of programming testing. This degree of testing is intended to uncover imperfections in the collaboration of coordinated units. Incorporation testing is helped by test pilots and test hits.
- Imperfections in the interfaces and collaborations between incorporated parts or frameworks are found through testing. Likewise see framework joining testing and part mix testing.

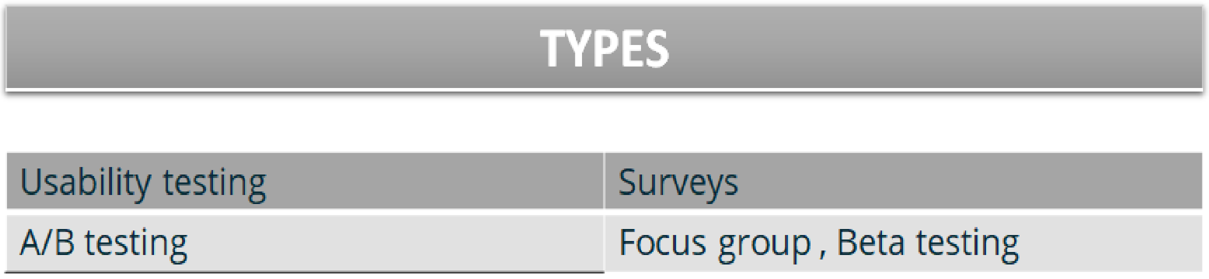


Unit / Component Integration Testing	System Integration Testing

Approaches	
Big Bang	Top Down
Bottom up	Hybrid

5.3 **User Testing**

User testing is the process of putting a website's, app's, product's, or service's interface and functions to the test by having real users do certain tasks in realistic situations.



TYPES	
Usability testing	Surveys
A/B testing	Focus group , Beta testing

5.4 **MC CALL's QUALITY FACTORS**

1. Product operation factors
    1. **Correctness**
        1. It is concerned with the accuracy of the software system's output.
        2. System gives correct accuracy and score for the different network traffic data.
    2. **Reliability**
        1. It is concerned with the failure of a service.
        2. System is reliable as it gives very minimal difference in the accuracy it has been provided.
    3. **Efficiency**
        1. It is worried about the equipment assets needed to do the product framework's different obligations.
        2. System uses very few resources to complete its task.
    4. **Integrity**
        1. This factor is concerned with the security of the software system, i.e., preventing unwanted access.
        2. System blocks access by unauthorized users to maintain security
    5. **Usability**
        1. It is concerned with the manpower required to train a new employee and operate the software system.
        2. System has a very good UI and is easy to use.

2. Product revision factors
    1. **Maintainability**
        1. This component takes into account the time and effort required by users and maintenance professionals to determine the causes of software problems.
        2. Systems do not require much maintenance even if used for a longer time.
    2. **Flexibility**
        1. This factor is concerned with the capabilities and effort required to support adaptive software maintenance activities.

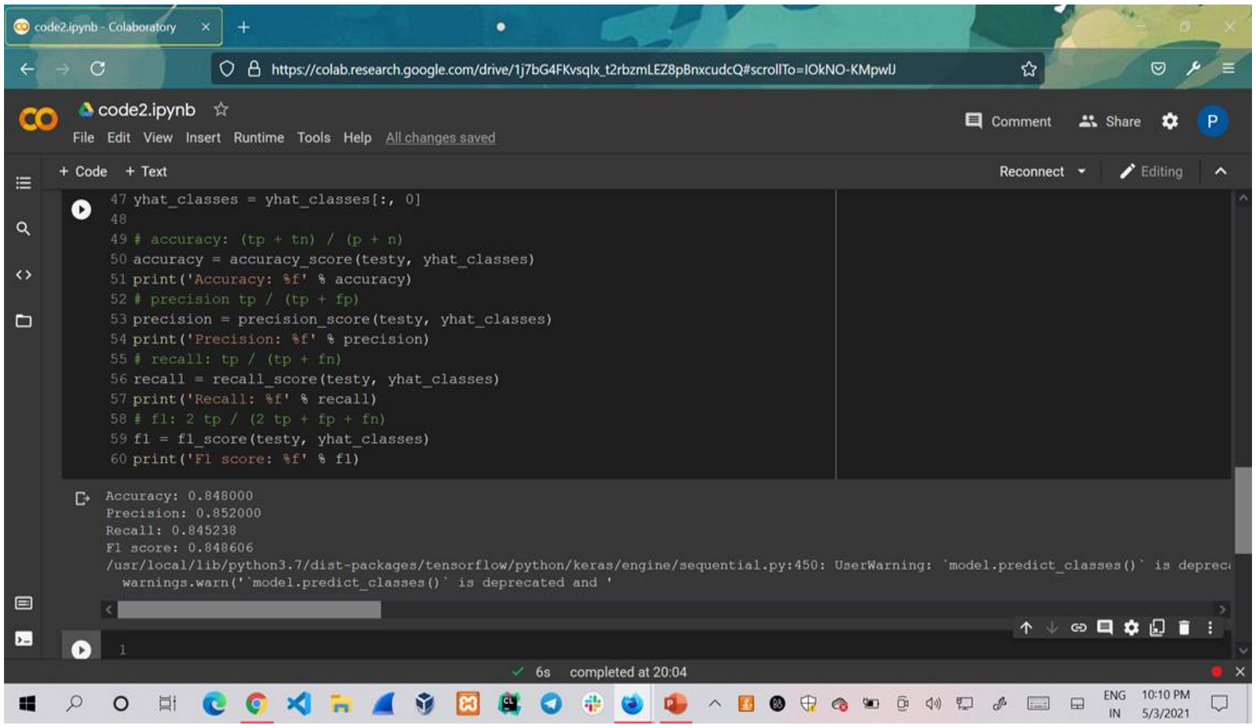
    2. Systems can easily adapt to new changes and features.

3. **Testability**
    1. Testability prerequisites are worried about both the testing and the working of the product framework.
    2. The System can be tested against each of its modules.

3. Product transition factors
    1. **Portability**
        1. It refers to a software system's adaptation to multiple surroundings with varied hardware and operating systems.
        2. The system is compatible to run easily in different environments.
    2. **Reusability**
        1. This component is worried about the utilization of programming modules made for a past project in a current programming project.
        2. The module of the system can be used in different modules with minor changes in it.
    3. **Interoperability**
        1. Its primary goal is to create interfaces with other software systems or equipment firmware.
        2. The normalized data output from one module can be used to input data in different modules very easily.

6.EXPERIEMENT RESULTS & ANALYSIS

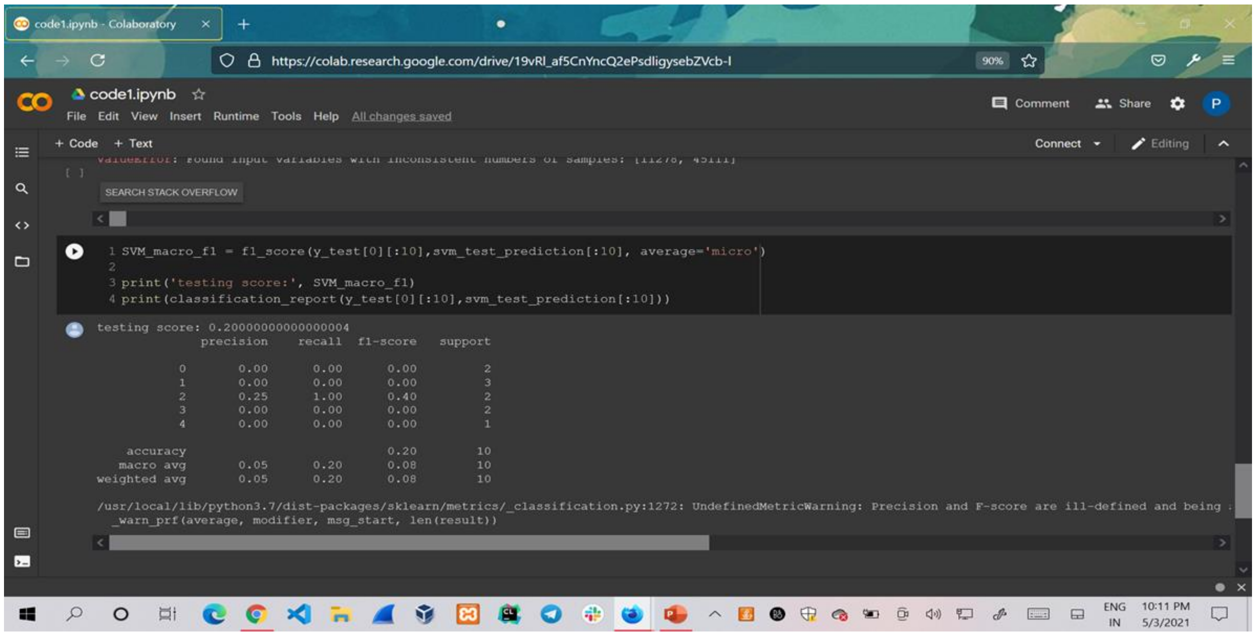
6.1 RESULT



```
yhat_classes = yhat_classes[:, 0]

# accuracy: (tp + tn) / (p + n)
accuracy = accuracy_score(testy, yhat_classes)
print('Accuracy: %f' % accuracy)
# precision tp / (tp + fp)
precision = precision_score(testy, yhat_classes)
print('Precision: %f' % precision)
# recall: tp / (tp + fn)
recall = recall_score(testy, yhat_classes)
print('Recall: %f' % recall)
# f1: 2 tp / (2 tp + fp + fn)
f1 = f1_score(testy, yhat_classes)
print('F1 score: %f' % f1)
```

```
Accuracy: 0.848000
Precision: 0.852000
Recall: 0.845238
F1 score: 0.848606
/usr/local/lib/python3.7/dist-packages/tensorflow/python/keras/engine/sequential.py:450: UserWarning: `model.predict_classes()` is depreca
  warnings.warn('`model.predict_classes()` is deprecated and '
```

Output: RNN

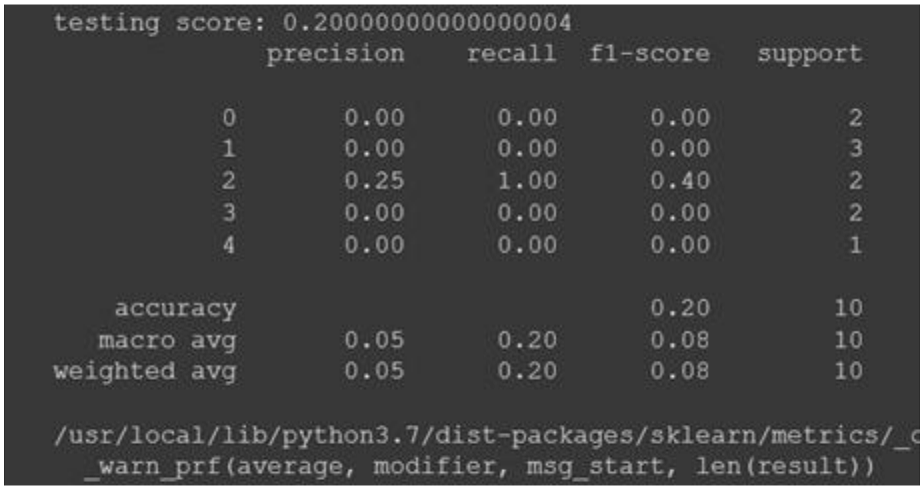


```
ValueError: Found input variables with inconsistent numbers of samples: [11278, 45111]
```

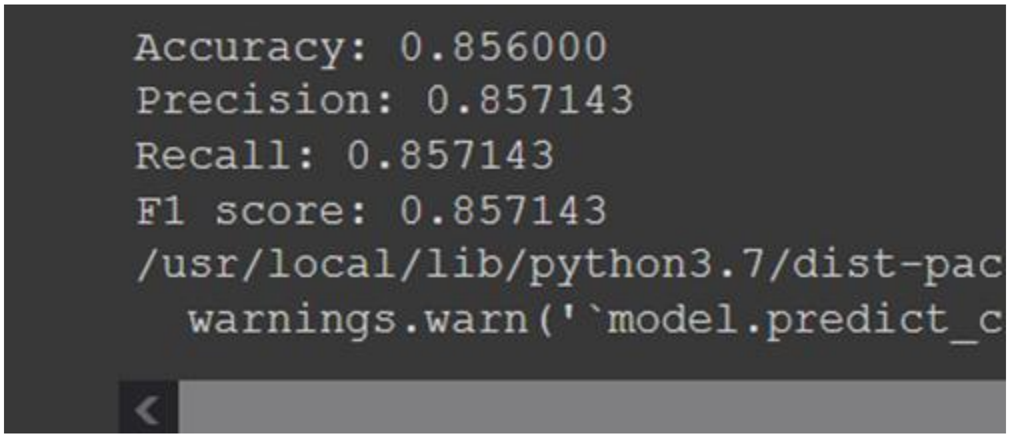
```
SVM_macro_f1 = f1_score(y_test[0][:10],svm_test_prediction[:10], average='micro')

print('testing score:', SVM_macro_f1)
print(classification_report(y_test[0][:10],svm_test_prediction[:10]))
```

```
testing score: 0.20000000000000004
              precision    recall  f1-score   support

           0       0.00      0.00      0.00         2
           1       0.00      0.00      0.00         3
           2       0.25      1.00      0.40         2
           3       0.00      0.00      0.00         2
           4       0.00      0.00      0.00         1

    accuracy                           0.20        10
   macro avg       0.05      0.20      0.08        10
weighted avg       0.05      0.20      0.08        10

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1272: UndefinedMetricWarning: Precision and F-score are ill-defined and being
  _warn_prf(average, modifier, msg_start, len(result))
```

Output: SVM



```
testing score: 0.20000000000000004
              precision    recall  f1-score   support

           0       0.00      0.00      0.00         2
           1       0.00      0.00      0.00         3
           2       0.25      1.00      0.40         2
           3       0.00      0.00      0.00         2
           4       0.00      0.00      0.00         1

    accuracy                           0.20        10
   macro avg       0.05      0.20      0.08        10
weighted avg       0.05      0.20      0.08        10

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_c
  _warn_prf(average, modifier, msg_start, len(result))
```



```
Accuracy: 0.856000
Precision: 0.857143
Recall: 0.857143
F1 score: 0.857143
/usr/local/lib/python3.7/dist-pac
  warnings.warn('`model.predict_c
```

We ran evaluation inspections on the SVM and RNN models utilizing datasets in this examination. We utilized exactness and large scale normal F1-score as execution measurements to outline the end-product concisely.

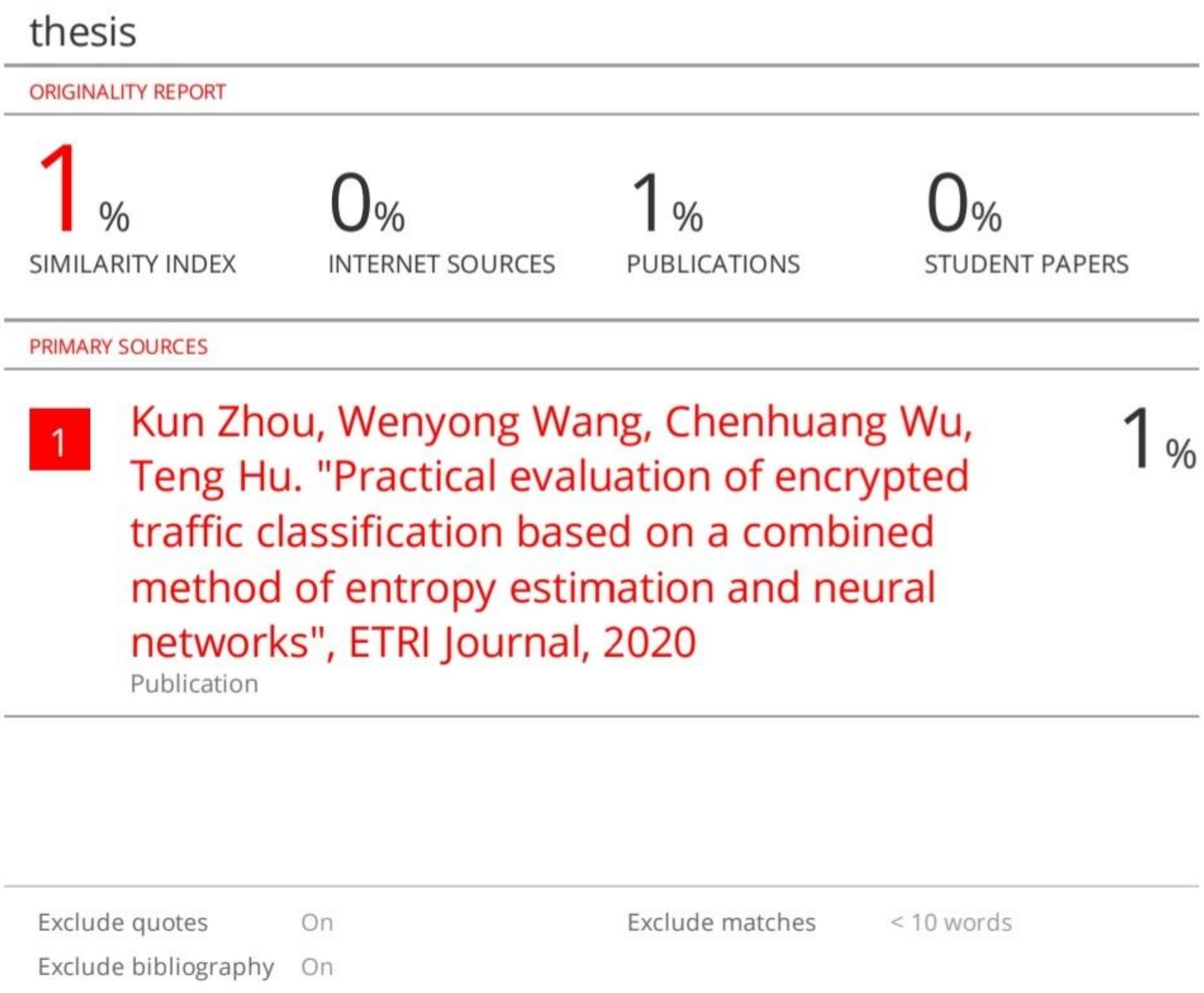
6.2 CONCLUSION & FUTURE WORK

The model demonstrates that SVM outperforms RNN in terms of accuracy.

This study proposes an encrypted visitor identity technique based entirely on the multilayer form. The experimental results show that the new strategy outperforms the existing strategy in terms of overall performance (SVM).

In the future, more community conditions and programmes will need to be considered in order to comprehend the suggested scheme's second stage.

7. PLAGIARISM REPORT



thesis

ORIGINALITY REPORT

1%	0%	1%	0%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Kun Zhou, Wenyong Wang, Chenhuang Wu, Teng Hu. "Practical evaluation of encrypted traffic classification based on a combined method of entropy estimation and neural networks", ETRI Journal, 2020 Publication	1%

Exclude quotes	On	Exclude matches	< 10 words
Exclude bibliography	On		

8. References

1. Capture Network Traffic Using Deep Neural Networks by Dimitra Chamou and Petros Toupas

https://onlinelibrary.wiley.com/doi/full/10.4218/etrij.2019-0190

2. Real Network Traffic Collection and Deep Learning for Mobile App Identification by Marco Picone

https://www.hindawi.com/journals/wcmc/2020/4707909/

3. An Encrypted Traffic Identification Scheme Based on the Multilevel Structure and Variational Automatic Encoder

https://www.hindawi.com/journals/scn/2020/8863169/

4. ENTROPY-BASED FEATURE EXTRACTION ALGORITHM FOR ENCRYPTED AND UNENCRYPTED COMPRESSED TRAFFIC CLASSIFICATION

http://www.ijicic.org/ijicic-150303.pdf

5. Pescape, A.: Entropy-based reduction of traffic data. IEEE Communications Letters 11(2), 191–193 (2007)

https://ieeexplore.ieee.org/document/4115159

6. Hjelmvik, E., John, W.: Breaking and improving protocol obfuscation. Tech. Rep. 2010-05, Computer Science and Engineering, Chalmers University of Technology (2010),

https://internetstiftelsen.se/docs/hjelmvik_breaking.pdf

7. Traffic Analysis Resistant Network (TARN) Anonymity Analysis
https://ieeexplore.ieee.org/document/8888134

8. A Semi-supervised Classification Algorithm for Encrypted Discrete Sequential Protocol Data Based on GAN

https://www.researchgate.net/publication/347962364_A_Semi-supervised_Classification_Algorithm_for_Encrypted_Discrete_Sequential_Protocol_Data_Based_on_GAN

9. Automatic Detection of Computer Network Traffic Anomalies based on Eccentricity Analysis

https://ieeexplore.ieee.org/document/8491507

10. Analysis and research of several network traffic prediction models
https://ieeexplore.ieee.org/document/6775859