~~Title~~

# PREDICTION OF ~~THIS PAPER~~MALICIOUS TRAFFIC IN IOT NETWORK

~~AUTHOR 1~~SPOORTHY REDDY JARUGU

Abstract. IoT devices are growing rapidly and due to there less computational power these devices are getting compromised easily. After the device is compromised attackers can easily enter the network though that device and gain access of the entire network. So it is necessary to collect the data when the device is affected and with the dataset gathered we should train the machine learning model. When such malicious communication happened model will predict it.

## Contents



*Date*: May 16, 2023.
2020 *Mathematics Subject Classification.* Artificial Intelligence.
*Key words and phrases.* Machine Learning, Data Mining, ...

## 1. ~~Introduction~~Problem Definition

~~At a high level, what is the problem area you are working in and why is it important? It is important to set the larger context here. Why is the problem of interest and importance to the larger community?~~

~~This paragraph narrows down the topic area of the paper~~Task is to predict malicious traffic in the IoT network. Prediction is done based on the labels that are assigned is the dataset. In the ~~first paragraph you have established general context and importance. Here you establish specific context and background.~~

~~"In this paper, we show that ...". This is the key paragraph in the intro - you summarize, in one paragraph, what are the main contributions of your paper given the context you have established in paragraphs 1 and 2. What is the general approach taken? Why are the specific results significant? This paragraph must be really good~~data generated have 22 columns where time, IP address for originating point and responding point, protocol used for communication, duration, connection state, no of packets, label are recorded. With the combination of all the columns finally labels says the particular communication happened is Benign or Malicious.

~~You should think about how to structure these one or two paragraph summaries of what your paper is all about.If there are two or three main results, then you might consider itemizing them with bullets or in test.~~

- ~~e.g., First ...~~
- ~~e.g., Second ...~~
- ~~e.g., Third . ..~~

~~If the results fall broadly into two categories, you can bring out that distinction here. For example, "Our results are both theoretical and applied in nature. (two sentences follow, one each on theory and application)"~~ Dataset is described below

Table 1. Dataset

| Dataset name | attributes |
|---|---|
| iot23˙combined˙new.csv | 'Unnamed: 0', 'ts', 'uid', 'id.orig˙h', 'id.orig˙p', 'id.resp˙h', 'id.resp˙p', 'proto', 'service', 'duration', 'orig˙bytes', 'resp˙bytes', 'conn˙state', 'local˙orig', 'local˙resp', 'missed˙bytes', 'history', 'orig˙pkts', 'orig˙ip˙bytes', 'resp˙pkts', 'resp˙ip˙bytes', 'label'], dtype='object |

~~Keep this at a high level, you can refer to a future section where specific details and differences will be given. But it is important for the reader to know at a high level, what is new about this work compared to other work in the area.~~

## 2. Data Preprocessing and visualization

~~"The remainder of this paper is structured as follows. .." Give the reader a roadmap for the rest of the paper. Avoid redundant phrasing, "In Section 2, In section 3, ... In Section 4, ... " etc.~~ Dataset contains 6046623 rows and 22 columns. We have to check for missing or NAN values and also check for the datatype of each column.

```
In [4]: dd.isnull().sum()

Out[4]: Unnamed: 0       0
        ts               0
        uid              0
        id.orig_h        0
        id.orig_p        0
        id.resp_h        0
        id.resp_p        0
        proto            0
        service          0
        duration         0
        orig_bytes       0
        resp_bytes       0
        conn_state       0
        local_orig       0
        local_resp       0
        missed_bytes     0
        history          0
        orig_pkts        0
        orig_ip_bytes    0
        resp_pkts        0
        resp_ip_bytes    0
        label            0
        dtype: int64
```

FIGURE 1. Checking for missing values

```
In [6]: dd.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6046623 entries, 0 to 6046622
Data columns (total 22 columns):
 #   Column         Dtype
---  ------         -----
 0   Unnamed: 0     int64
 1   ts             float64
 2   uid            object
 3   id.orig_h      object
 4   id.orig_p      float64
 5   id.resp_h      object
 6   id.resp_p      float64
 7   proto          object
 8   service        object
 9   duration       object
 10  orig_bytes     object
 11  resp_bytes     object
 12  conn_state     object
 13  local_orig     object
 14  local_resp     object
 15  missed_bytes   float64
 16  history        object
 17  orig_pkts      float64
 18  orig_ip_bytes  float64
 19  resp_pkts      float64
 20  resp_ip_bytes  float64
 21  label          object
dtypes: float64(8), int64(1), object(13)
memory usage: 1014.9+ MB
```

FIGURE 2. Data type of each column

In the process of cleaning the dataset we have seen that 'uid', 'ts', 'Unnamed: 0' are having unique values so we are removing it.

```
In [9]: # We can see that uid and ts have unique value so we can remove it
        dd=dd.drop(columns=['uid','ts','Unnamed: 0'])
```

FIGURE 3. Dropping the values

~~Test citation [1]. and [2] or Beliakov et al. [2].~~ Using IP address in network traffic dataset as a feature can be used for detecting anomalies or identifying network patterns. With the help of the below code we are converting it to int data type which is easy for machine learning to process.
~~This is for , and this is for .~~

Converting orginator IP address string to its corresponding integer representation

```
In [12]: ip_col_name = "id.orig_h"

         # define a function to convert IP addresses to integers
         def ip_to_int(ip):
             try:
                 return struct.unpack("!I", socket.inet_aton(ip))[0]
             except socket.error:
                 return None

         dd[ip_col_name] = dd[ip_col_name].apply(ip_to_int)

         dd.to_csv("iot23_combined_new.csv", index=False)
```

FIGURE 4. Converting Ip address to int

~~Number: . , , , and~~ Coming to 'Service', 'duration', 'orig_bytes', 'resp_bytes' when using values_counts( ) we can see that there are some values which is represented with special character '_' that have to replaced.

```
In [20]: dd['service']=dd['service'].str.replace('-','nil')
```

FIGURE 5. Service

~~We have , , the range: . 1/2.~~

```
In [25]: dd['duration']=dd['duration'].str.replace('-','0')
```

FIGURE 6. Duration

```
In [27]: dd['orig_bytes']=dd['orig_bytes'].str.replace('-','0')
```

FIGURE 7. Originator bytes

```
In [29]: dd['resp_bytes']=dd['resp_bytes'].str.replace('-','0')
```

FIGURE 8. Respondent bytes

~~For , as shown below:~~ Conn state: Connection state of the network traffic is displayed in pi chart. Where we can see 'S0' and 'OTH' occupies the maximum portion.

$$a = b \times \sqrt{ab}$$

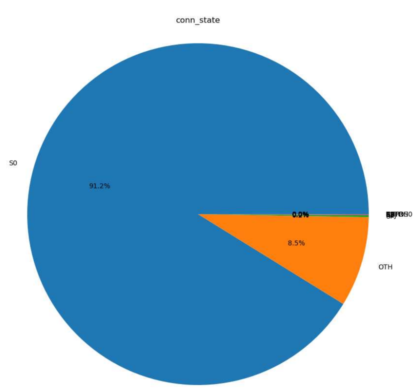


FIGURE 9. Connection state pi chart representation

To understand the split up of other values we have ignored those two values and represented it in bar chart for remaining values.

## 3. ~~PRELIMINARIES~~

FIGURE 10. Connection state filtered bar chart

- Label is the main feature in our dataset which is used for prediction. this feature says weather the network traffic is malicious or benign.
- Below is the pi chart representation of label feature.

## 3. ~~Method~~

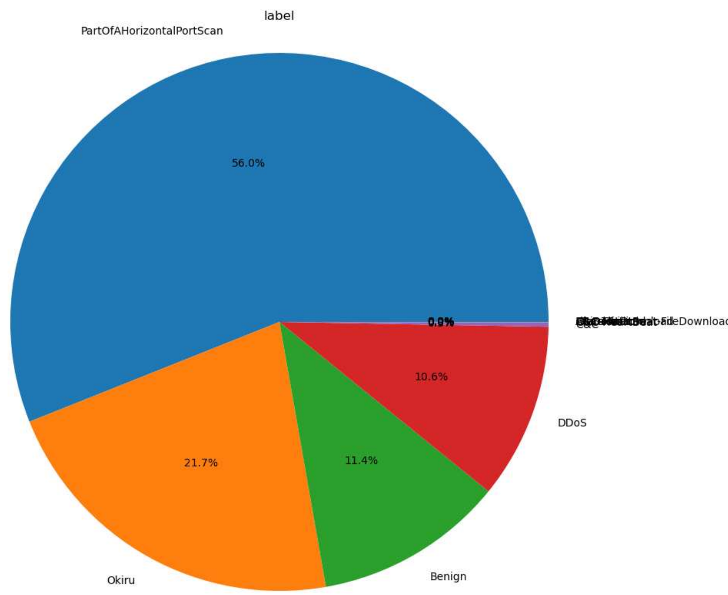


Figure 11. Pi chart representation of label

## 3. ~~Experiment and Analysis~~

### ~~Precision Comparison on Event Detection Methods~~

Table 2. Label

| Name | ~~OR Event Detection~~ attributes |
|---|---|
| Label | ~~AC Event Detection~~ PartOfAHorizontalPortScan |
| | ~~TC Event Detection~~ Okiru |
| ~~precision~~ | ~~0.83~~ Benign |
| | ~~0.69~~ DDoS |
| | ~~0.46~~ C&C |
| ~~recall~~ | ~~0.68~~ C&C-HeartBeat |
| | ~~0.48~~ Attack |
| | ~~0.36~~ C&C-FileDownload |
| ~~F-score~~ | ~~0.747~~ C&C-Torii |
| | ~~0.57~~ FileDownload |
| | ~~0.4~~ C&C-HeartBeat-FileDownload |
| | Okiru-Attack |
| | C&C-Mirai |

Label feature have 13 values where the majority is occupied by 4 values as shown above. Removing those four values in below image we represent the rest of the values in bar chart.

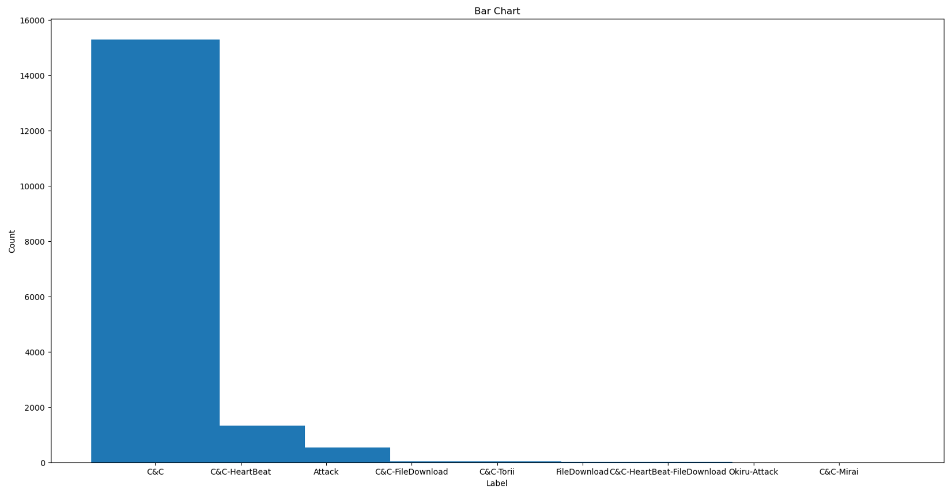


Figure 12. Filtered label representation

## 3. Label Encoding

Categorical data are those that are represented by labels or categories, such as gender, color, or type of product. Machine learning algorithms require numerical input, so numerical encoding is necessary to transform categorical data into numerical values. We use label encoding to convert 'proto', 'service', 'conn_state', 'label' variables to numerical values.

**Label Encoding**

```
In [49]: pmap = {'tcp':0,'udp':1,'icmp':2}
         dd['proto']=dd['proto'].map(pmap)

In [50]: pmap = {'nil':0,'dns':1,'irc':2,'http':3,'dhcp':4,'ssl':5,'ssh':6}
         dd['service']=dd['service'].map(pmap)

In [51]: pmap = {'S0':0,'OTH':1,'SF':2,'REJ':3,'S3':4,'RSTR':5,'SH':6,'RSTO':7,'RSTOS0':8,'S1':9,'SHR':10,'S2':11,'RSTRH':12}
         dd['conn_state']=dd['conn_state'].map(pmap)

In [52]: pmap = {'PartOfAHorizontalPortScan':0,'Okiru':1,'Benign':2,'DDoS':3,'C&C':4,'C&C-HeartBeat':5,'Attack':6,'C&C-FileDownload':7,'C&
         dd['label']=dd['label'].map(pmap)
```

Figure 13. Label encoding

Correlation is a statistical measure that describes the relationship between two variables. It is used to determine how strongly and in what way two variables are related to each other. Correlation coefficients range between -1 and +1. We are representing correlation using heat map.

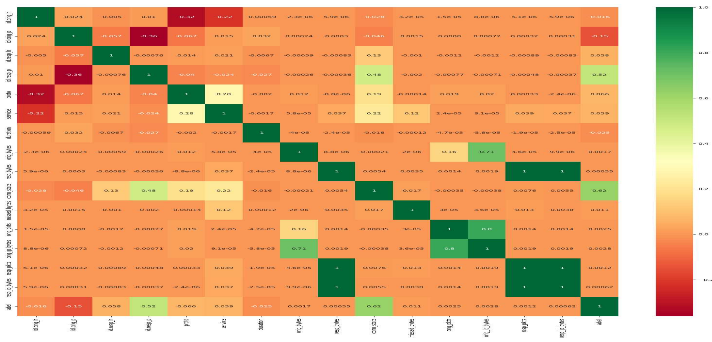

FIGURE 14. Correlation

If variables are highly correlated, it may be a good idea to remove one of the variables to avoid multicollinearity, which can lead to unstable and inaccurate models. When we refer the above diagram we can see that some variables are highly correlated. To avoid multicollinearity we are removing 'resp_bytes', 'resp_pkts', 'orig_pkts'.

## 4. Model built and prediction

The most common way to split a dataset is to divide it into two subsets: a training set and a testing set. The training set is used to train the model, and the testing set is used to evaluate its performance. Training set is of 80% and test is of 20%.

```
In [56]: X = dd.drop(['label'], axis=1)
         y = dd[['label']]
```

FIGURE 15. Splitting the value to train and test set

The decision tree algorithm starts with a root node that represents the entire dataset. It then recursively splits the dataset into subsets based on the values of one of the input features, and creates a decision node for each split. This process continues until a stopping criterion is met, such as reaching a certain depth or purity level. We used Decision Tree Classifier which is the supervised learning algorithm at the end it prints out important feature with its values.

```
In [93]: # Build the decision tree
         df = DecisionTreeClassifier()
         df.fit(X_train, y_train)

         # Evaluate the decision tree
         y_pred = df.predict(X_test)
         accuracy = accuracy_score(y_test, y_pred)

         importances = df.feature_importances_

         feature_names = list(X.columns)

         feature_importances = sorted(zip(feature_names, importances), key=lambda x: x[1], reverse=True)

         for feature, importance in feature_importances:
             print(f"{feature}: {importance}")
```

FIGURE 16. Decision tree classifier

- Calculates the feature importance of the model
- Sorts the features by importance in descending order
- Prints each feature and its importance

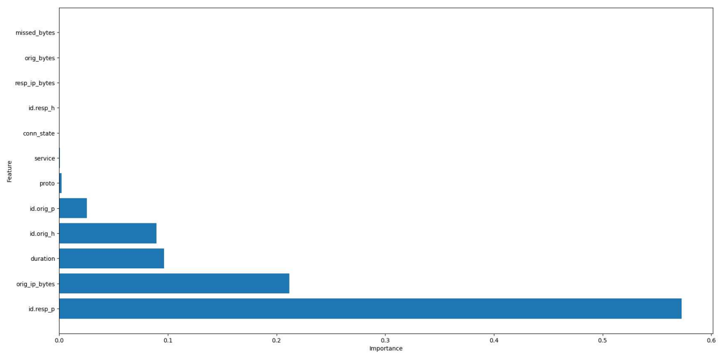


FIGURE 17. Important features used

## 5. EVALUATING THE MODEL

Evaluating machine learning models - Accuracy: This is a measure of the proportion of correct predictions made by the model. It is calculated as the number of correct predictions divided by the total number of predictions. However, accuracy can be misleading in cases where the classes are imbalanced.

```
In [99]: mse = mean_squared_error(y_test, y_pred)
         print("MSE:", mse)

         MSE: 8.434457238542162e-05

In [100]: rmse = mean_squared_error(y_test, y_pred, squared=False)
          print("RMSE:", rmse)

          RMSE: 0.009183930116536254

In [101]: mae = mean_absolute_error(y_test, y_pred)
          print("MAE:", mae)

          MAE: 2.6461042316995017e-05

In [103]: accuracy = accuracy_score(y_test, y_pred)
          print("Accuracy:", accuracy)

          Accuracy: 0.9999875963864139
```

FIGURE 18. Accuracy obtained

- Confusion matrix is used to evaluate the performance of a classification model. It compares the actual and predicted target variables and counts the number of correct and incorrect predictions

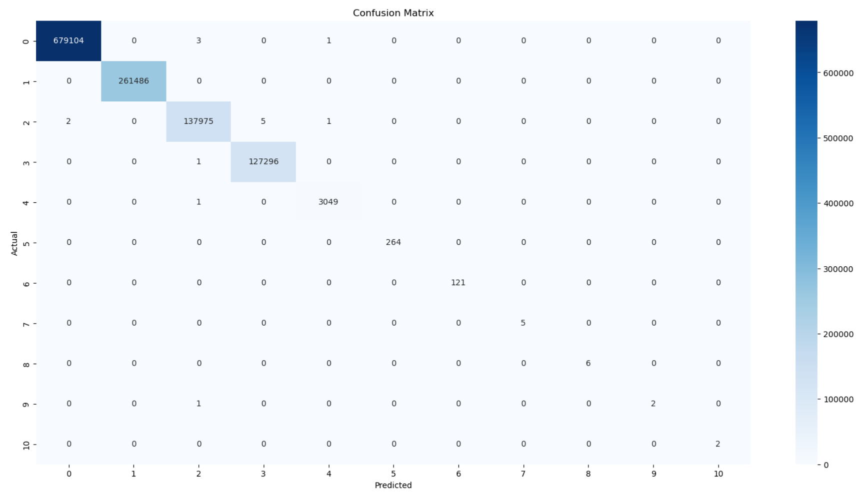


FIGURE 19. Evaluation result

## 6. CONCLUSIONS

- For this Flip00 task I have taken dataset from Kaggle. For which data pre-processing, data cleaning, splited the data for train and test, finally applied Decision Tree Classifier algorithm.
- Additionally used originator IP address (id.orig_h) and respondent IP address (id.resp_h) by converting them to integer form.

~~ACKNOWLEDGEMENT~~

- This prediction model helps to effectively predict the Labels of traffic.

~~The authors would like to thank ...~~

## References

[1] Gleb Beliakov and Gang Li. Improving the speed and stability of the k-nearest neighbors method. *Pattern Recognition Letters*, 33(10):1296–1301, 2012.

[2] Gleb Beliakov, Simon James, and Gang Li. Learning choquet-integral-based metrics for semisupervised clustering. *Fuzzy Systems, IEEE Transactions on*, 19(3):562–574, 2011.

## List of Todos

(A. 1) School of Computer Science and Engineering,, Vellore Institute of Technology, Vellore, India