National College of Ireland

Configuration Manual

MSc Research Project
Cybersecurity

Shreyas Srinivasa
Student ID: X23102641

School of Computing
National College of Ireland

Supervisor: Dr. Arghir Moldovan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Shreyas Srinivasa
Student ID:	23102641
Programme:	Cybersecurity **Year:** 2024-2025
Module:	MSc Research Project
Lecturer:	Dr. Arghir Moldovan
Submission Due Date:	12-12-2024
Project Title:	In-Depth Analysis of Machine Learning for Securing Internet of Things devices using CiC IoT & Net Flow Dataset
Word Count:	..... **Page Count:** .....

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

Signature: .....

Date: 12/12/2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

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

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

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

Configuration Manual

Shreyas Srinivasa
X23102641

Introduction

This document describes the step-by-step procedure to execute the research project and gives in detail information about the configurations which is required to run the models with the UQ-NIDS, CICIoT dataset.

Experimental Setup

System Configuration

Hardware Used in this Experiment	Version	Purpose
HP Pavilion 14dv1001TU Processor: 8-Core Intel Core i5 Memory: 16 GB	Windows 11	Workstation

Software Used in this Experiment

Google Colab is the free servi,ce hosted by Google for programming, education/ and research. The code requires the following libraries: NumPy, Pandas, Plotly, Matplotlib, Seaborn, Scikit-learn, Tensor Flow, Keras.

Dataset

The newly publishes datasets which are UQ-NIDS 2023 and CICIoT 2023. The UQ-NIDS dataset has 11,994,893 records, out of which 9,208,048 (76.77%) are benign flows and 2,786,845 (23.23%) attacks. Whereas the CICIoT 2023 contains over 548 GB of raw network traffic data which are stored in PCAP format. Malicious traffic has 33 different attack types which are categorized into 7 primary groups. There is a total of 46,686,579 records. DDoS category contains 33,984,560 records, DoS category comprises of 8,090,738 records which focuses on single source attacks which can disrupt the services availability. PSHACK flood comprises of 4,094,755, HTTP flood comprises of 28,790 records. SYN flood comprises of 2,028,834 records. The spoofing category has 486,504 records. And Mirai Botnet category consists of 2,634,124 records, Benign Traffic has 1,098,195 records.

UQ-NIDS source: https://staff.itee.uq.edu.au/marius/NIDS_datasets/
CICIoT source: https://www.unb.ca/cic/datasets/iotdataset-2023.html

1) Upload CSV file to Google Drive. The size of CICIoT dataset is 12.8GB and the size of UQ-NIDS is 490MB.
2) Mount the Drive to Colab.

```
from google.colab import drive
drive.mount('/content/drive')
!ls "/content/drive/MyDrive/CiC-DataSet/Complete_Dataset/csv"
```

```
Mounted at /content/drive
 CICIoT2023  'README_csv - README.pdf'
```

```
data_features = pd.read_csv('/content/drive/MyDrive/Project/NetFlow_v1_Features.csv')
raw_data = pd.read_csv('/content/drive/MyDrive/Project/NF-UQ-NIDS.csv')
```

Model Training

For CICIoT dataset, The program processes each training set file using tqdm for progress tracking.
The dataset files are split into training sets (80%) and test sets (20%).
The chunks are processed using scaler.partial_fit(chunk). This line suggests a missing scaler definition, likely meant to be MinMaxScaler() or StandardScaler(). This scaling process ensures that feature values are normalized for model training. The program takes 8 hours to completely execute as the dataset is ver vast and the processing time for each model is complex and our research provides in depth analysis to figure out which model works best for the dataset and for which type of attack configuration.

For the UQ-NIDS dataset, program prints the content of the data_features DataFrame, lik.ely showing a list of features used in the network intrusion detection task. These features might describe traffic patterns, proto.cols, flags, and statistical measures the dimensions of the dataset, showing the number of rows and columns. This helps un;derstand the dataset's size before preprocessing and training. Network Attacks: Denial of Service, Reconnaissance, Scanning, etc, Exploitation Attacks: Exploits, Backdoor, Botnets, etc., Benign: Non-malicious traffic. And Combines all Network Attacks and Exploitation Attacks.
Selects a corresponding numb.er of Benign rows to ensure balance, setting the total to 100,000 samples. The Features (X): All columns except the target Attack_Category.
Target (y): The Attack_Category column.

Implementation

The CICIoT dataset is trained for four models they are Logistic Regression, Decision Tree, Random Forest and Support Vector Machine. The attack categories are classified into three groups 34 class (33+1) classification, 8 class (7+1) classification, 2 class (1+1) classification.

The UQ-NIDS dataset is trained for Decision Tree, Random Forest, Logistic Regression and Naïve Bayes machine learning models and along with these models, we have implemented hyperparameter tuning technique in order to fine tune the model and increase its accuracy.

Model Implementation

The UQ-NIDS dataset had the following output to the implemented machine learning language.

```
Random Forest Model Accuracy: 97.12%
Random Forest Model Precision: 0.9715
Random Forest Model Recall: 0.9712
Random Forest Model F1_score: 0.9712
```

```
XGBoost Model Accuracy: 97.08%
XGBoost Model Precision: 0.9718
XGBoost Model Recall: 0.9708
XGBoost Model F1_score: 0.9709
```

Model (For **UQ-NIDS**)	Accuracy	Precision	Recall	F1 Score
Decision Tree	96.24	97.15	97.13	97.13
Random Forest	97.12	97.15	97.12	97.12
XG Boost	97.08	97.18	97.26	97.09
LogReg	64.59	65.65	64.59	64.67
Naive Bayes	49.47	56.37	49.47	46.00

The models were were subjected to hyper parameter tuning, they produced similar results except for Naïve Bayes (55.95%) which produced significant improvements compared to without hyperparameter tuning. And LogReg (68.87%) performed slightly better with hyperparameter tuning.

Model (With Hyper Parameter Tuning)	Accuracy	Precision	Recall	F1 Score
Decision Tree	96.74	96.74	96.74	96.74
Random Forest	97.04	97.04	97.12	97.12
XG Boost	97.08	97.18	97.08	97.09
LogReg	68.87	68.96	68.87	68.3
Naive Bayes	55.95	61.96	55.95	55.68

```
XGBoost Model Accuracy: 97.08%
XGBoost Model Precision: 0.9718
XGBoost Model Recall: 0.9708
XGBoost Model F1_score: 0.9709
```

```
Naive Bayes Model Accuracy: 55.95%
Naive Bayes Model Precision: 0.6196
Naive Bayes Model Recall: 0.5595
Naive Bayes Model F1_score: 0.5568
```

```
Decision Tree Model Accuracy: 96.74%
Decision Tree Model Precision: 0.9674
Decision Tree Model Recall: 0.9674
Decision Tree Model F1_score: 0.9674
```

```
Logistic Regression Best Model Accuracy: 68.87%
Logistic Regression Best Model Precision: 0.6896
Logistic Regression Best Model Recall: 0.6887
Logistic Regression Best Model F1_score: 0.6830
```

The CICIoT dataset is a large dataset which comprises of 169 files and has storage of 12.8 GB, so computing such large datasets takes much time and we implemented additional GPUs for our research so as to compute the models without any hurdle.

For extensive research the CICIoT dataset has been split into three different categories based on the attack categories. 34 class (33+1) , 8 class (7+1) and 2 class (1+1) classifications.

CICIoT 2023 - 34 Class				
Model	Accuracy	Precision	Recall	F1 Score
Decision Tree	99.1981	80.6247	81.0326	80.6382
Random Forest	96.55	96.55	96.44	96.44
LogReg	80.1831	59.4978	48.528	49.1284
SVM	78.7127	52.8674	42.7667	43.3722

For 34 class classification, Decision Tree produced the highest acuuracy with 99.19% while Support Vector Machine model produced the lowest acuuracy with 78.71% which suggests that SVM is the least performing model.

```
##### DecisionTree #####
Accuracy Score:  0.9919811693454289
Recall Score:  0.8103269213100808
Precision Score:  0.8062475944896136
F1 Score:  0.80638206531345
```

```
##### Evaluation Results #####
Accuracy Score =  0.8230735478876973
Recall Score =  0.46898625413482875
Precision Score =  0.6774741457451166
F1 Score =  0.5017935247751959
```

CICIoT 2023 - 8 Class				
Model	Accuracy	Precision	Recall	F1 Score
Decision Tree	99.4054	83.1231	82.757	82.9336
Random Forest	96.55	96.55	96.44	96.44
LogReg	83.147	51.0881	68.3171	53.7237
SVM	82.3073	67.7474	46.8986	50.1793

For 8 class classification, Decision Tree produced the highest acuuracy with 99.40% while Support Vector Machine model produced the lowest acuuracy with 82.30% which suggests that SVM is the least performing model.

```
##### LogisticRegression (8 classes) #####
Accuracy Score =  0.8314706124982
Recall Score =  0.6831711618042706
Precision Score =  0.5108819896711103
F1 Score =  0.537237291622334
```

```
##### DecisionTree #####
Accuracy Score:  0.9919811693454289
Recall Score:  0.8103269213100808
Precision Score:  0.8062475944896136
F1 Score:  0.80638206531345
```

CICIoT 2023 - 2 Class				
Model	Accuracy	Precision	Recall	F1 Score
Decision Tree	99.5888	95.5463	95.5038	95.525
Random Forest	96.55	96.55	96.44	96.44
LogReg	98.902	86.3226	89.0443	87.6315
SVM	98.7115	87.1835	83.5327	85.2584

For 2 class classification, Decision Tree produced the highest acuuracy with 99.5% while Support Vector Machine model produced the lowest acuuracy with 98.71% which suggests that SVM is the least performing model.

```python
from sklearn.metrics import accuracy_score, recall_score, precision_score, f1_score

y_pred = preds[0]

# Print evaluation metrics
print("##### Decision Tree (2 classes) #####")
print('accuracy_score: ', accuracy_score(y_test, y_pred))
print('recall_score: ', recall_score(y_test, y_pred, average='macro'))
print('precision_score: ', precision_score(y_test, y_pred, average='macro'))
print('f1_score: ', f1_score(y_test, y_pred, average='macro'))
print()
```

```
##### Decision Tree (2 classes) #####
accuracy_score:  0.9958889421547692
recall_score:  0.9550381776930854
precision_score:  0.9554638197525818
f1_score:  0.9552508939534676
```

```python
from sklearn.metrics import accuracy_score, recall_score, precision_score, f1_score
for k,v in preds.items():
    y_pred = v
    print(f"##### {ML_names[k]} (2 classes) #####")
    print('Accuracy Score: ', accuracy_score(y_pred, y_test))
    print('Recall Score: ', recall_score(y_pred, y_test, average='macro'))
    print('Precision Score: ', precision_score(y_pred, y_test, average='macro'))
    print('F1 Score: ', f1_score(y_pred, y_test, average='macro'))
    print()
    print()
    print()
```

```
##### LogisticRegression (2 classes) #####
Accuracy Score:  0.9890278304177276
Recall Score:  0.890443912103491
Precision Score:  0.8632265127120491
F1 Score:  0.8763159471404506
```