Configuration Manual

MSc Research Project
Cyber Security

Arindam Ghoshal
Student ID: 20194587

School of Computing
National College of Ireland

Supervisor: Vikas Sahni

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Arindam Ghoshal
Student ID:	20194587
Programme:	MSc in Cyber Security **Year:** 2021-2022
Module:	MSc Internship
Lecturer:	Vikas Sahni
Submission Due Date:	07/01/2022
Project Title:	Intrusion detection in Industrial OT environment by combination of different machine learning techniques
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: .....

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):	

Appendix H – Monthly Internship Activity Report

The Internship Activity Report is a 1-page monthly summary of the activities performed by you and what you have learned during that month. The Internship Activity Report must be signed off by your Company and included in the configuration manual as part of the portfolio submission.

Student Name: Arindam Ghoshal Student number: 20194587

Company: Ornua co-operative Ltd Month Commencing: Oct 2021

Role Description:
The aim of the internship was to study and understand the role of Intrusion Detection System in Operational Technology and further researching on the ways to enhance the detection rate of intrusion in the OT environment using Machine learning techniques.

1. Studied and analyzed the current Operation Technology Threat Detection models.
2. Carried out the research for the Cyber-attack detection in OT environment.
3. Prepared the documentation on the related research work studied.

Employer comments

Arindam Ghoshal has performed all the internship tasks mentioned above sincerely and successfully.

Student Signature: DocuSigned by: Arindam Ghoshal 06E8BCC53800408... Date: 30/10/2021

Industry Supervisor Signature: Declan Smith Date: 30/10/2021

Appendix H – Monthly Internship Activity Report

The Internship Activity Report is a 1-page monthly summary of the activities performed by you and what you have learned during that month. The Internship Activity Report must be signed off by your Company and included in the configuration manual as part of the portfolio submission.

Student Name: Arindam Ghoshal Student number: 20194587

Company: Ornua co-operative Ltd Month Commencing: Nov 2021

The task performed are:

1. Prepared the documentation on the related research work studied.
2. Analyzed historical methodologies and came up with methodology for the proposed solution using machine learning algorithms such as Random Forest and Convolution Neural Network.

Employer comments

- Arindam carried out the assigned tasks in time successfully.

Student Signature: DocuSigned by: Arindam Ghoshal 06E8BCC35800408... Date: 30/11/2021

Industry Supervisor Signature: Declan Smith Date: 30/11/2021

Appendix H – Monthly Internship Activity Report

The Internship Activity Report is a 1-page monthly summary of the activities performed by you and what you have learned during that month. The Internship Activity Report must be signed off by your Company and included in the configuration manual as part of the portfolio submission.

Student Name: Arindam Ghoshal Student number: 20194587

Company: Ornua co-operative Ltd Month Commencing: Dec 2021

The task performed are:

1. Performed development activity for proposed solution.
2. Developed and evaluated the models.
3. Prepared the documentation for the activity performed

Employer comments

- Arindam carried out the assigned tasks in time and dedicatedly.

Student Signature: DocuSigned by: Arindam Ghoshal 06E8BCC35800408... Date: 20/12/2021

Industry Supervisor Signature: Declan Smith Date: 20/11/2021

Configuration Manual

Arindam Ghoshal
Student ID: 20194587

1 Introduction

The configuration Manual describes the requirements to create the research environment and displays the implementation steps including the necessary hardware, software and snippets of codes used for completing the research work. The purpose of this manual is to provide step wise activities undertaken for the project and it would act as a handbook to replicate or simulate the undertaken project to verify the results in future.
This project was focussed on selection of optimised model for prediction of intrusions in the industrial operations technology environment with the help of machine learning models working in isolation and in hybrid or combined modes.

2 Implementation

2.1 Hardware

This section provides the details of the hardware used to support the research experiments using machine learning algorithm. Analyses of huge datasets with machine learning accompanies high resource utilization and hence below information would be helpful in providing an idea of resource requirement.

- CPU: AMD Ryzen 5 4600H with clock speed of 3.00 GHz
- Memory: RAM 8GB DDR4
- Storage: 512 GB SSD

2.2 Software, tools, and connectivity

- Integrated Development Environment (IDE): Google Colaboratory (Google colab)
- Coding language: Python 3.7
- Data verification tool: Microsoft excel
- Data storage: PC/ Google Drive
- Connectivity: Stable internet connectivity for using Google colab (cloud-based IDE)
- Data sampling tool: RapidMiner

3 Dataset

The dataset used for the project consists of events related to electric transmission system simulated in the lab of Mississippi State University and Oak Ridge National Laboratory (Borges Hink *et al.*, 2014).The dataset includes events related to

- synchro phasor measurements and

- data logs from control panels and relays

<bound method NDFrame.head of R1-PA1:VH R1-PM1:V R1-PA2:VH ... snort_log3 snort_log4 marker

0	70.399324	127673.0908	-49.572308	...	0	0	Natural		
1	73.688102	130280.7109	-46.300719	...	0	0	Natural		
2	73.733939	130305.7842	-46.254883	...	0	0	Natural		
3	74.083443	130581.5902	-45.899649	...	0	0	Natural		
4	74.553268	131083.0556	-45.424094	...	0	0	Natural		
...	...	...	...	...	...	...	...		
5271	141.589330	131885.4002	21.606238	...	0	0	Natural		
5272	141.526305	131960.6200	21.566131	...	0	0	Natural		
5273	141.365877	132035.8398	21.394244	...	0	0	Natural		
5274	141.348688	132035.8398	21.371326	...	0	0	Natural		
5275	141.251285	132035.8398	21.291112	...	0	0	Natural		

[78377 rows x 129 columns]>

Fig 1: Snapshot of dataset used for this project.

4 Datafiles used for the analyses

The files which have been used for this project are mentioned below.

- *20194587_DissertationF.ipynb*: This the configuration or coding file which was loaded into google colaboratory IDE for execution.
- *Data1 to Data15*: These are available raw dataset, which was processed further for analyses.
- *Final_df_unsampled (1).csv*: This is the collated, sampled and balanced dataset used for analyses with the help of different ML models.

5 Python Libraries

The research project used Python as the coding language to configure and run the analyses models and hence different python libraries were imported in google colaboratory - IDE for various functionalities which were used in this research.

List of Libraries:

- Pandas
- Numpy
- Sklearn
- Mlxtend.classifier
- RandomForestClassifier
- K-NN classifier
- MLP classifier
- Stacking classifier
- confusion_matrix
- accuracy_score
- seaborn
- matplotlib

```
import pandas as pd
import numpy as np

import seaborn as sns
import matplotlib.pyplot as plt
import keras

from sklearn.model_selection import train_test_split

from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import confusion_matrix,accuracy_score,roc_auc_score,roc_curve
from sklearn.preprocessing import LabelBinarizer
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.neural_network import MLPClassifier

from mlxtend.classifier import StackingClassifier
import six
import sys
sys.modules['sklearn.externals.six'] = six
```

Fig 1: Snapshot of libraries used for this project.

Pandas: Pandas is a Python library for machine learning. Pandas is a free and open-source programming library.
Pandas is often used for data analytics and loading datasets. Pandas is used for machine learning in several domains, including finance, economics, and others. It is very user-friendly and can convert datasets to a tabular format for easier comprehension.

NumPy: NumPy is a Python library for machine learning. In Python, NumPy is used to manage arrays. NumPy is used to do any computations using 1- or 2-dimensional arrays. Additionally, NumPy has functions that do linear algebra and Fourier transform operations.

Sklearn: It is one of the most essential Python libraries for machine learning. Sklearn includes a number of tools for statistical modelling, classification, regression, clustering, and dimension reduction.
Matplotlib: This is a library for data visualization. It is a Python open-source module for plotting graphs from model results. These charts may aid in comprehending the context of the findings. Various components of the findings may be visually presented to aid with comprehension.

Seaborn: Seaborn is a library for data visualization in Python that is developed on top of matplotlib and tightly coupled with pandas' data structures. Seaborn's key component is visualization, which aids in the exploration and comprehension of data.

6 Data Pre-processing

Data processing included several steps which are mentioned below.

6.1 Step 1

Individual datasets were collected and then merged into a final single dataset.

```
alldataframes = [df1, df2, df3, df4, df5, df6, df7, df8, df9, df10, df11, df12, df13, df14, df15]
merged_df = pd.concat(alldataframes)
```

```
print(merged_df.shape)
```

```
(78377, 129)
```

Fig 2: Individual datasets merged to form collected dataset.

```
cln_merged_df = merged_df.dropna()
cln_merged_df.shape
```

```
(78377, 129)
```

```
print(merged_df.isna().sum())
```

```
R1-PA1:VH     0
R1-PM1:V      0
R1-PA2:VH     0
R1-PM2:V      0
R1-PA3:VH     0
              ..
snort_log1    0
snort_log2    0
snort_log3    0
snort_log4    0
marker        0
Length: 129, dtype: int64
```

Fig 3: Merged dataset was cleaned of any null values.

6.2 Step 2

Dataset was balanced with equal amount of malicious and benign events to avoid any sort of overfitting. Here RapidMiner tool was used to up sample the natural(benign) data to equal level of attack data.

```
sns.countplot(x='marker',data=merged_df, palette='hls')
plt.show()
plt.savefig('count_plot')
```

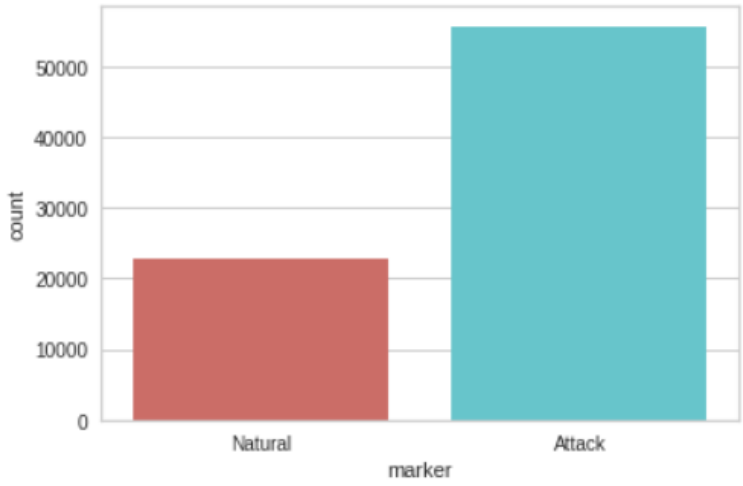

Fig 4: Un-balanced dataset

Fig 5: RapidMiner up-sampling 'natural' events.

```
sns.countplot(x='marker',data=final_df, palette='hls')
plt.show()
plt.savefig('count_plot')
```

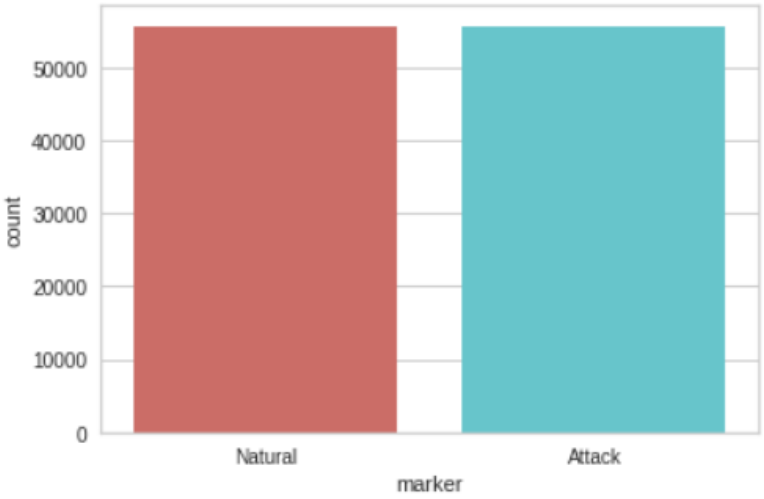

Fig 6: Balanced dataset.

6.3 Step 3

Unwanted and null values were removed from the dataset which might prevent classifiers from working as desired.

```
final_df = final_df.dropna()
final_df.shape
```

```
(101983, 129)
```

Fig 7: Null values were dropped.

6.4 Step 4

The string values were replaced by numerical values.

```
cleanup_nums = {"marker": {"Natural": 0, "Attack": 1} }
final_df = final_df.replace(cleanup_nums)
final_df.shape
```

```
(111326, 129)
```

Fig 8: String values converted to numerical values.

6.5 Step 5

Dataset was divided into training and testing datasets further in 70% and 30% ratio.

```
X = final_df.iloc[:, final_df.columns != 'marker']
y = final_df.iloc[:,-1:]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)

print("X-train shape:", X_train.shape)
print("y-train shape:", y_train.shape)
print("X-test shape:", X_test.shape)
print("y-test shape:", y_test.shape)
```

```
X-train shape: (71388, 128)
y-train shape: (71388, 1)
X-test shape: (30595, 128)
y-test shape: (30595, 1)
```

Fig 9: Dataset segmented into train and test sets.

7 Classification Models

This research work consists of 4 classification models using machine learning algorithms given below.

- Random Forest
- K-Nearest Neighbor

- Multilayer Perceptron
- Stacked Ensemble Learning

7.1 Random Forest classifier

Random forest classifier was used in isolation to find out its performance on the dataset and it proved to be fair over most K-NN and MLP.

```
###### Random Forest

rfclf = RandomForestClassifier().fit(X_train, y_train)

rf_y_test_pred = rfclf.predict(X_test)
confusion_matrix(y_test, rf_y_test_pred)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:3: DataConversionWarning: A column-vector y w
  This is separate from the ipykernel package so we can avoid doing imports until
array([[14232,   910],
       [  786, 14667]])
```

```
print("Random Forest :: Accuracy on Training: ", rfclf.score(X_train,y_train)*100)
```

```
Random Forest :: Accuracy on Training:  100.0
```

```
print("Random Forest :: Accuracy on Test: ", rfclf.score(X_test,y_test)*100)
```

```
Random Forest :: Accuracy on Test:  94.45661055728061
```

Fig 10: Random Forest classifier code snippet.

7.2 K-NN Classifier

K-NN classifier was run in isolation over the dataset, and it proved to be fair over MLP but failed to succeed over other two models.

```
###### K-nearest neighbour

knnclf = KNeighborsClassifier(n_neighbors=3).fit(X_train,y_train)

knn_y_test_pred = knnclf.predict(X_test)
confusion_matrix(y_test,knn_y_test_pred)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-vect
  return self._fit(X, y)
array([[12343,  2799],
       [ 1865, 13588]])
```

```
print("KNN :: Accuracy on Training: ", knnclf.score(X_train,y_train)*100)
```

```
KNN :: Accuracy on Training:  92.34185017089706
```

```
print("KNN :: Accuracy on Test: ", knnclf.score(X_test,y_test)*100)
```

```
KNN :: Accuracy on Test:  84.75567903252166
```

Fig 11: K-NN classifier code snippet.

7.3 Multilayer perceptron (MLP) Classifier

MLP classifier is an unsupervised classifier, and it was engaged in analyses of the dataset in isolation. The result did not stand fair to other models and accuracy could reach up to 53 % at the most.

```
from sklearn.neural_network import MLPClassifier
dt = MLPClassifier()
dt.fit(X_train,y_train)
y_pred = dt.predict(X_test)
print("MLP:Confusion Matrix: ", confusion_matrix(y_test, y_pred))
print("MLP:Accuracy : ", accuracy_score(y_test,y_pred)*100)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py:1109: DataConversionWarning: A co
  y = column_or_1d(y, warn=True)
MLP:Confusion Matrix:  [[12507  2635]
 [12359  3094]]
MLP:Accuracy :  50.99199215558098
```

Fig 12: MLP classifier code snippet.

7.4 Stacked Ensemble Classifier

Stacked ensemble classifier is a hybrid classifier which takes output of other classifiers to compute the best performance and deliver it as result. All classifier described above were used as input for stacked ensemble classifier in this project and the final performance was computed, which stood higher that isolated classification outputs.

```
#Stacking
stk.fit(X_train,y_train)
stk2 = StackingClassifier(Classifiers2,meta_classifier=stk)
stk2.fit(X_train,y_train)
y_pred = stk.predict(X_test)
print("Stacking:Confusion Matrix: ", confusion_matrix(y_test, y_pred))
print ("Stacking:Accuracy : ", accuracy_score(y_test,y_pred)*100)


confusion_matrix(y_test,knn_y_test_pred)
```

```
/usr/local/lib/python3.7/dist-packages/mlxtend/classifier/stacking_classification.py:154: DataConversionWarning: A column-vector
  clf.fit(X, y)
/usr/local/lib/python3.7/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-vector y was pas
  return self._fit(X, y)
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py:1109: DataConversionWarning: A column-vec
  y = column_or_1d(y, warn=True)
/usr/local/lib/python3.7/dist-packages/mlxtend/classifier/stacking_classification.py:154: DataConversionWarning: A column-vector
  clf.fit(X, y)
/usr/local/lib/python3.7/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-vector y was pas
  return self._fit(X, y)
Stacking:Confusion Matrix:  [[14204   938]
 [  800 14653]]
Stacking:Accuracy :  94.31933322438307
array([[12343,  2799],
       [ 1865, 13588]])
```

Fig 13: Stacked ensemble classifier code snippet.

8 Evaluation

Python code was used to calculate the statistical outputs of the classifiers mentioned above and graphs and matrices were produced.

```
#confusion Matrix
matrix =confusion_matrix(y_test,y_pred)
class_names=[0,1]
fig, ax = plt.subplots()
tick_marks = np.arange(len(class_names))
plt.xticks(tick_marks, class_names)
plt.yticks(tick_marks, class_names)
sns.heatmap(pd.DataFrame(matrix), annot=True, cmap="YlGnBu" ,fmt='g')
ax.xaxis.set_label_position("top")
plt.tight_layout()
plt.title('Confusion matrix', y=1.1)
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
plt.show()
```

Fig 14: Snippet of python code for deriving confusion matrix

References

Borges Hink, R.C. *et al.* (2014) 'Machine learning for power system disturbance and cyber-attack discrimination', in *2014 7th International Symposium on Resilient Control Systems (ISRCS). 2014 7th International Symposium on Resilient Control Systems (ISRCS)*, pp. 1–8. doi:10.1109/ISRCS.2014.6900095.