Research Configuration Manual

MSc (Research)
Practicum Part 2

Tushar Gharpure
Student ID: x23289902

School of Computing

Masters in data Analytics

National College of Ireland

Supervisor: Shubham Shubhnil

National College of Ireland Project Submission Sheet School of Computing

Student Name:	Tushar Gharpure
Student ID:	x23289902
Programme:	MSc (Research) Practicum Part 2
Year:	2025
Module:	MSc Research Project
Supervisor:	Shubham Shubhnil
Submission Due Date:	11/08/2025
Project Title:	Research Configuration Manual
Word Count:	1301
Page Count:	9

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:	11th August 2025

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):	

Research Configuration Manual

Tushar Gharpure x23289902

1. Introduction

This manual will explain how to set up and run our code. Here the code is comparing three deep learning models and at last saving the best performing model for web application for detect of real time cyber attacks like Dos, DDos, Brute Force, Web-Based attack and scans.

Configuration Manual

2. System Configutations

2.1 Hardware Configurations

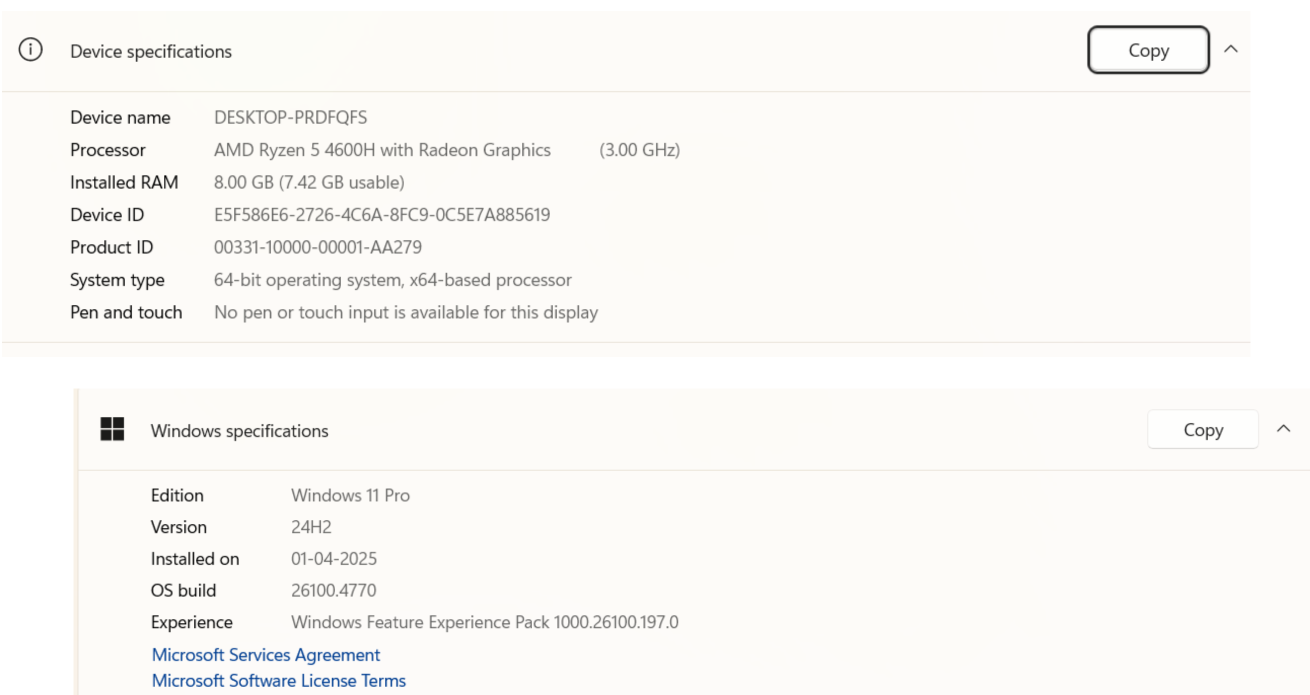

Figure 1: Hardware configurations

2.2 Software Configurations

It is the most important section of the implementation. This section will be highlighting important software requirements for our research.

- Python Version: 3.9 or later
- Platform: Windows / Linux / macOS
- RAM: At least 4GB
- Editor: VS Code / Terminal / Google Colab

For Colab

2.2.1 Google Colab

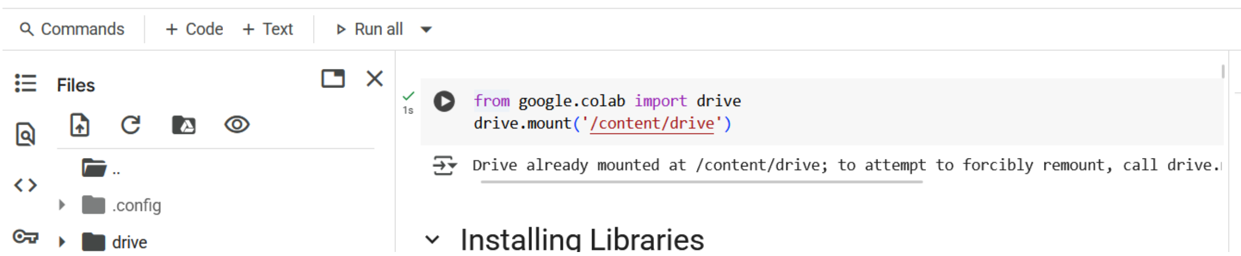

Fig 2: Accessing google Drive on Colab

This colab is use to implement the code in this project, it a service from google which provides free computing resources such as GPU and CPU for running the python code. All required libraries are preloaded in google colab already and here the dataset is saved on google drive then we mount that dataset to colab for further implementations.

This is the One-Time Setup for Colab.

1. Upload Tushar_IoT_code.ipynb to Colab.

2. Mount Google Drive

3. Ensure the folder /content/drive/MyDrive/iot_dataset/ exists and contains the CSVs and (if predicting) scaler.pkl + conv_lstm_model.keras.

Here for better performance we can switch the google colab from CPU to GPU_or TPU for better runtime for much faster results. This can be done by changing the runtime.

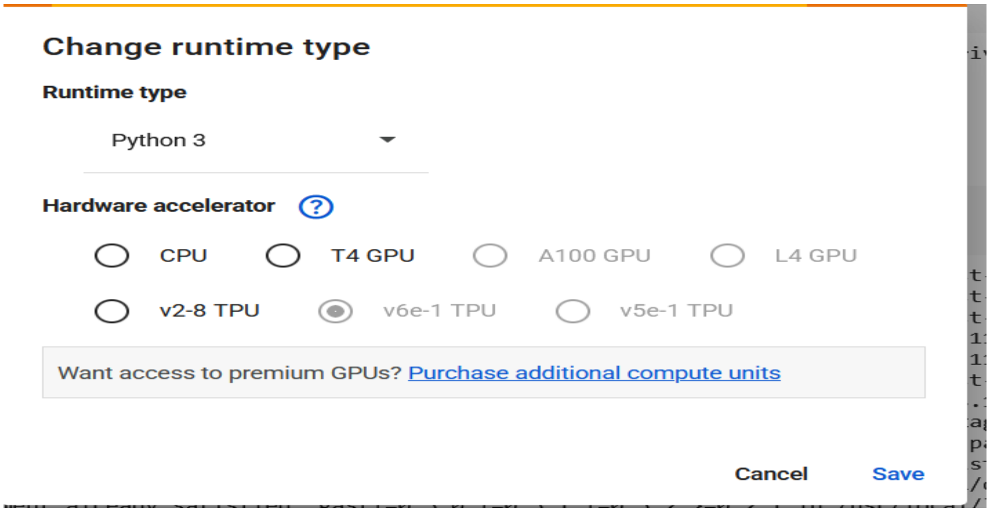

Figure 3: GPU Runtime In Google Colab

3. Data Preparation

Here the dataset is in CSV format and it consists of two features flow based and packet based, both are in csv format.

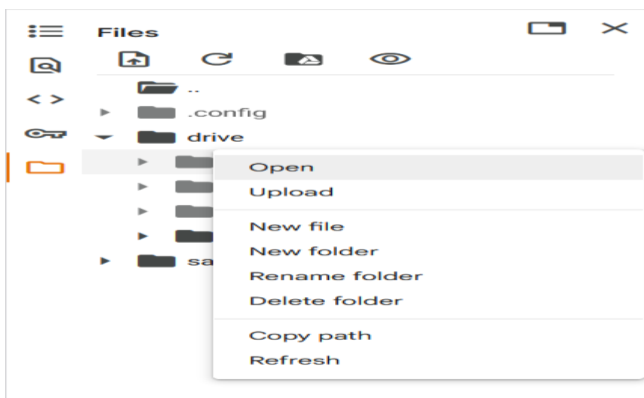

Figure 4: Upload of Datasets

From here we can upload our dataset, our dataset is in folder which is having two features for each traffic.

4. Importing all required libraries

```python
# Core Libraries
import numpy as np
import pandas as pd
import textwrap
from collections import Counter
import joblib

# Visualization
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go

# Scikit-learn (Data Prep & Evaluation)
from sklearn.impute import SimpleImputer
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix

# Imbalanced-learn
from imblearn.over_sampling import SMOTE

# TensorFlow / Keras
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import (
    Conv1D, MaxPooling1D, Flatten, Dense, Dropout, BatchNormalization,
    LSTM, TimeDistributed
)
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import EarlyStopping
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.regularizers import l2
from tensorflow.keras.models import load_model
```

Figure 5: Importing all required libraries.

This is all the important libraries that is essential for this project after that the next steps are Data Pre-processing which includes checking nulls and duplicates values), EDA(used to define patterns and class imbalance of data), Temporal analysis (showing attack timing in a day based on hours), Feature Engineering(extracting feature and selecting features).

5. Saving best performing model

Saving the Best Performing Model

```python
conv_lstm_model.save("/content/drive/MyDrive/iot_dataset/conv_lstm_model.keras")
```

The best performing model is saved and further used on real time data for the detection of threats.

6. Structure of code(sequence)

a. Mounting of data
b. Installing Libraries and importing them
c. Loading of dataset
d. Merging of data key features(flow based and packed based)
e. Merging in big DataFrame(all_attacks_df)
f. Data Preprocessing includes checking duplicates, checking missing values and missing columns, handling missing values and dropping duplicate columns
g. Checking data types.

h. Exploratory data analysis (identifying distribution and classes and finding data imbalance)
i. Temporal analysis which the timestamp of attack
j. Feature engineering where timestamp is extracted into hour, min, dayofweek etc.
k. Dropping columns with having only one unique values
l. Applying LabelEncoder
m. Using of RandomForrest for selection 95% of important data
n. Saving test csv for further real time threat detection.
o. Saving scaler which is having scaled features.
p. Applying model – CNN
q. Applying – LSTM
r. Applying – Conv – LSTM
s. Generating evaluation report for three models
t. Comparing Accuracy, Precision, Recall and F1-score
u. Saving the best performing model for real time threat detection on web-app
v. Real Time Anomaly detection on web-app.

Epochs – While implementing models 10 epochs will give best accuracy.

conv_lstm_model.keras – This file we have to download from colab and manually save in web-app model folder directory.

scaler.pkl – This file we have to download from colab and manually save in web-app model folder directory.

X_test_for_prediction.csv - This file we have to download from colab and manually save in web-app input folder directory.

7. Troubleshooting for most occurring errors are as below.

- **ModuleNotFoundError**
  Re-run the pip install cell and then Restart runtime and re-run.

- **TensorFlow version / GPU issues**
  Use Colab default TF or install tensorflow in a fresh environment (Python 3.10/3.11). GPU is optional.

- **FileNotFoundError**
  Check Drive mount and path spelling: /content/drive/MyDrive/iot_dataset/.... Upload missing CSVs/artifacts.

- **Shape/feature mismatch at predict time**
  Your input must be scaled with the same scaler.pkl and must follow the exact feature order used in training.

For Web-App

1. System Requirements

- Python Version: 3.9 or later
- Platform: Windows / Linux / macOS
- RAM: At least 4GB
- Editor: VS Code / Terminal / Google Colab

2. Required Python Libraries

Use the following pip commands to install the required libraries:

- pip install flask
- pip install tensorflow
- pip install pandas
- pip install numpy
- pip install scikit-learn

3. Folder Structure

The folder structure after extracting the ZIP file is as follows:

final_webapp_iot/

|**---app.py** # Flask web app (main entry point)

|**---readme.txt** # Readme instructions (basic)

|**---input**

|**---X_test_for_prediction.csv** # Sample input data (preprocessed IoT data)

|**---model**

|**--- conv_lstm_model.keras** # Pre-trained Conv-LSTM model

|**---scaler.pkl** # Scaler object used during training

|**---templates**

|**--- index.html** # Frontend HTML interface

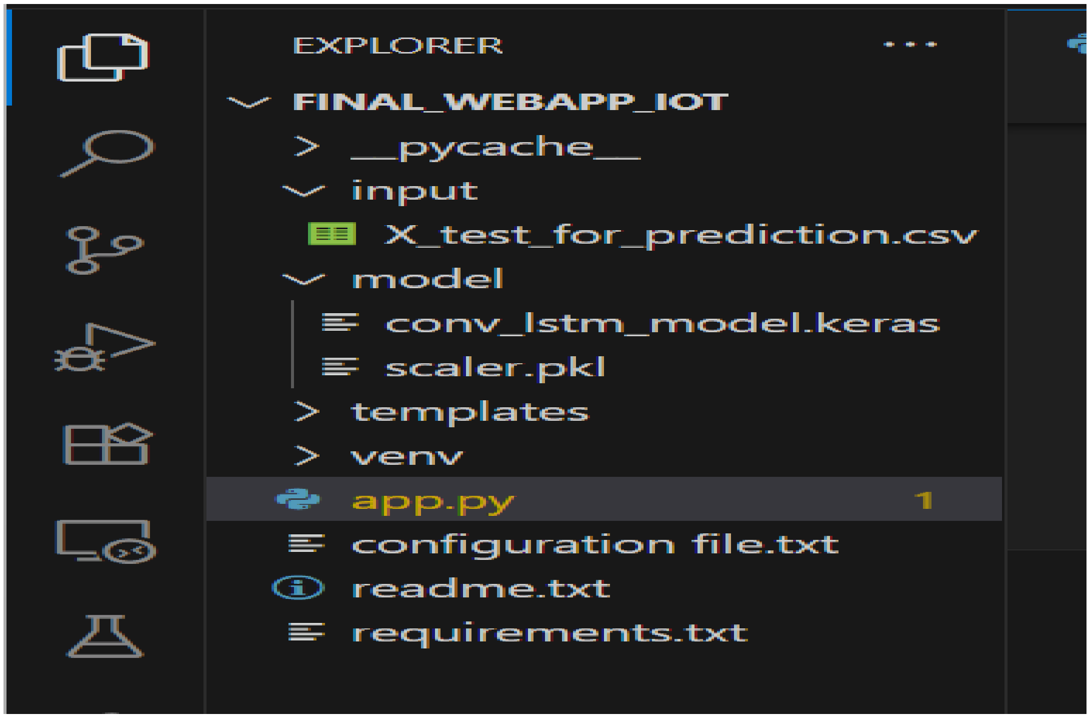

Figure 6: Folder Directory for WebApp

4. How to Run the Web App

Step 1: Extract the ZIP

Unzip final_webapp_iot.zip into a working folder.

Step 2: Create Virtual Environment

- python -m venv venv
- venv\Scripts\activate # Windows
- source venv/bin/activate # Linux/macOS

Step 3: Install Required Libraries

Run the following command:

- pip install flask tensorflow pandas scikit-learn

Step 4: Launch the Web App

- python app.py

Open your browser and visit: http://127.0.0.1:5000/

5. Notes for Testing

- Ensure the input CSV structure matches the training data format.
- Uploaded CSV is normalized using the scaler.pkl.

6. Common Errors & Fixes

- Error: Shape mismatch or ValueError

Ensure the CSV is reshaped into the correct (samples, timesteps, features) format.

- Error: ModuleNotFoundError

Install the missing library using pip.

- Error: Model not loading

Check that the conv_lstm_model.keras and scaler.pkl files are in the correct directory.

References:

- Abadi, M. et al. (2016) 'TensorFlow: A system for large-scale machine learning', *Proceedings of OSDI '16*, pp. 265–283.
- Chollet, F. (2015) *Keras*. Available at: keras.io (Accessed: 11 August 2025).
- Pedregosa, F. et al. (2011) 'Scikit-learn: Machine learning in Python', *Journal of Machine Learning Research*, 12, pp. 2825–2830.