National College of Ireland

Configuration Manual

MSc Research Project
MSc in Data Analytics

Gayathri Gangadharan
Student ID: X22203427

School of Computing
National College of Ireland

Supervisor: Aaloka Anant

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Gayathri Gangadharan
Student ID:	X22203427
Programme:	MSc in Data Analytics
Year:	2024
Module:	MSc Research Project
Supervisor:	Aaloka Anant
Submission Due Date:	12/12/2024
Project Title:	Configuration Manual
Word Count:	877
Page Count:	5

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:	Gayathri Gangadharan
Date:	12th December 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

Gayathri Gangadharan
X22203427

1 Introduction

This configuration manual provides guidelines on how prepare and execute short-term(hourly) and long-term(daily and monthly) traffic flow prediction models, Artificial Neural Network(ANN) and Convolutional Neural Network (CNN), using Dublin's traffic dataset from the Sydney Coordinated Adaptive Traffic System(SCATS). The manual aims to replicate all the project conditions to ensure accurate predictions and thorough analysis.

2 System and Software requirements

This research was done on a Windows 10 desktop with 8 GB of RAM,sufficient for data preprocessing and training the model.The implementation was performed in Python using Jupiter Notebook, with Anaconda employed for managing python environment.The required libraries for this project are TensorFlow, Keras, Pandas, NumPy, Scikit-learn, Matplotlib, and Seaborn.

3 Data Source

For this research, Dublin's traffic dataset from SCATS for the month of July 2024 was used. It provides historical traffic flow records for Dublin, accessed through the link: `https://data.smartdublin.ie/dataset/dcc-scats-detector-volume-jul-dec-2024`.

4 Data Pre-processing

The dataset was loaded and pre-processed using the Pandas library in python,which included handling null values and transforming variable types.

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

```
# Load dataset
df1=pd.read_csv("SCATSJuly2024.csv")
```

Figure 1: Data loading

5 Feature Engineering

As the first step in feature engineering, Label encoding was applied to convert the categorical variables into numerical formats as shown in figure 2. The next step is Data splitting, the dataset was split into features and target variables, followed by further division into training and testing sets as shown in figure 3.Additionally, data scaling was performed to normalize the features and target variable, bringing them to a common scale for improved model efficiency. These tasks were executed using 'LabelEncoder()', 'train_test_split()', and 'StandardScaler' classes from the Scikit-learn library.

```
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()

# Apply the label encoder to the 'Region' column
df1['Region'] = label_encoder.fit_transform(df1['Region'])
# Check the transformed DataFrame
print(df1.head())
print(df1['Region'].unique())
```

Figure 2: Label encoding

```
from sklearn.model_selection import train_test_split

# Define target variable and feature set
X = df1[['Region', 'Site', 'Detector', 'Avg_Volume', 'Year', 'Month', 'Day', 'Hour']]
y = df1['Sum_Volume']

# Split into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Figure 3: Data Splitting

```
from sklearn.preprocessing import StandardScaler

# Select features to scale
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Figure 4: Data Scaling

6 Model Training

Two deep learning models ANN and CNN were selected for this study because of their capability to capture non-linear patterns in traffic data and identify temporal and spatial patterns which are crucial for accurate predictions.In learning the feature interactions

ANN is more proficient, and CNN, on the other hand, its convolutional layer is capable of extracting the important patterns both in the time and space domain, this makes both models relevant to traffic flow prediction at hourly, daily and monthly basis.

```python
import tensorflow as tf
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Dense, Dropout

# Build the ANN model
model = Sequential([
    Dense(64, activation='relu', input_shape=(X_train.shape[1],)),
    Dropout(0.2),
    Dense(32, activation='relu'),
    Dropout(0.2),
    Dense(1)  # Single output for regression task
])

# Compile the model
model.compile(optimizer='adam', loss='mse', metrics=['mae'])
```

(a) ANN Model Creation

```python
from tensorflow.keras.layers import Conv1D, MaxPooling1D, Flatten
```

```python
# Reshape data to fit into a CNN model
X_train_cnn = X_train.reshape((X_train.shape[0], X_train.shape[1], 1))
X_test_cnn = X_test.reshape((X_test.shape[0], X_test.shape[1], 1))

# Build the CNN model
cnn_model = Sequential([
    Conv1D(32, kernel_size=2, activation='relu', input_shape=(X_train_cnn.shape[1], 1)),
    MaxPooling1D(pool_size=2),
    Flatten(),
    Dense(64, activation='relu'),
    Dropout(0.2),
    Dense(1)  # Single output for regression task
])
```

(b) CNN Model creation

Figure 5: Model Creation

For monthly traffic flow prediction, the models were developed in python using Keras package from TensorFlow library as shown in figure 5a and 5b.

For daily prediction, the data was aggregated by grouping it based on attributes like region,site, detector, year, month, and day and for hourly prediction the data was aggregated by grouping it based on attributes like region,site, detector, year, month, hour and day (refer figure 6).The feature, 'Avg_Volume', representing the mean of traffic volume and 'Sum_Volume', representing the cumulative traffic volume for each day.Then, it was divided into training and testing sets using 'train_test_split()' function from Scikit-learn library.

```python
# Aggregate data for hourly predictions
df_hourly = df1.groupby(['Region', 'Site', 'Detector', 'Year', 'Month', 'Day', 'Hour']).agg({
    'Sum_Volume': 'sum',
    'Avg_Volume': 'mean'
}).reset_index()
```

```python
# Aggregate data for daily predictions
df_daily = df1.groupby(['Region', 'Site', 'Detector', 'Year', 'Month', 'Day']).agg({
    'Sum_Volume': 'sum',
    'Avg_Volume': 'mean'
}).reset_index()
```

Figure 6: Data Aggregation

The ANN and CNN models were implemented for daily and hourly prediction.The architecture of ANN model, comprising multiple dense layers with ReLU activation and dropout for regularization, was the same that used form monthly predictions(refer figure 5a).The model was trained over 10 epochs with a batch size of 32, utilizing the test dataset for validation.Following training, the model was evaluated on the test set to generate hourly and daily traffic flow predictions.
The CNN model for daily and hourly data followed the same architecture and implementation approach as shown in the CNN model for monthly prediction(refer figure 5b).It began with a convolutional layer of same filters and size and ReLU activation.The model consisted of a max-pooling layer, followed by a flattening layer, a dense layer, a dropout layer, and a final regression layer.It was compiled with Adam optimizer and trained for 10 epochs with validation data to assess performance (refer figure 7).

```
model.fit(X_train, y_train, epochs=10, batch_size=32, validation_split=0.2, verbose=1)
```

Figure 7: Model training

ANN and CNN models were created for a specific region 'IRE', selected from the seven region's in the dataset, to predict traffic flow at hourly, daily and monthly intervals.The region 'IRE' was filtered from the dataset (refer figure 8), with the columns divided into features(Region, site, detector, year,month,day, and hour) and the target variable (Sum_volume).Then the ANN and CNN models were implemented to predict traffic flow at hourly, daily and monthly intervals.

Prediction for a specific location - IRE

```
: #Filter data for Region 2 (assuming Region 2 is encoded as '2')
  df_region2 = df1[df1['Region'] == 2]
```

Figure 8: Filter data for 'IRE'

7 Model Testing and Evaluation

```
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
```

```
mae = mean_absolute_error(y_test, predictions)
mse = mean_squared_error(y_test, predictions)
rmse = np.sqrt(mse)
r2=r2_score(y_test,predictions)

print(f"R-squared: {r2}")
print(f'Mean Absolute Error: {mae:.4f}')
print(f'Mean Squared Error: {mse:.4f}')
print(f'Root Mean Squared Error: {rmse:.4f}')
```

Figure 9: Model evaluation

The ANN and CNN models were evaluated to assess their accuracy and reliability using standard metrics such as R2 values, MSE, MAE and RMSE, by importing packages

'r2_score','mean_squared_error', 'mean_absolute_error' from the Scikit-learn library and the RMSE is calculated using the NumPy library as shown in figure 9 .The model's predictions were compared with actual values to determine their suitability for short-term and long-term forecasting.Predictions were made using the trained models and analyzed for accuracy across different time intervals: hourly, daily and monthly.