ConfigurationManual

MScResearchProject
DataAnalytics

ValarineElizabethRoyappa
Student ID: X21231605

SchoolOfComputing
NationalCollegeOfIreland

Supervisor: Mr.ArjunChikkankod

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Valarine Elizabeth Royappa
Student ID:	X21231605
Programme:	Data Analytics
Year:	2022
Module:	MSc Research Project
Supervisor:	Mr Arjun Chikkankod
Submission Due Date:	18/09/2023
Project Title:	Configuration Manual
Word Count:	592
Page Count:	6

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:	18th September 2023

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

Valarine Elizabeth Royappa
X21231605

1 Introduction

We will see all the deployed techniques used and the hardware applications used for the "Genetic Algorithm Based Sentiment Analysis for Cyberbullying Detection" in this con@iguration guide.

2 System & Software Specification

The system setup used for this research is depicted in Figure 1 along with the software specs used.

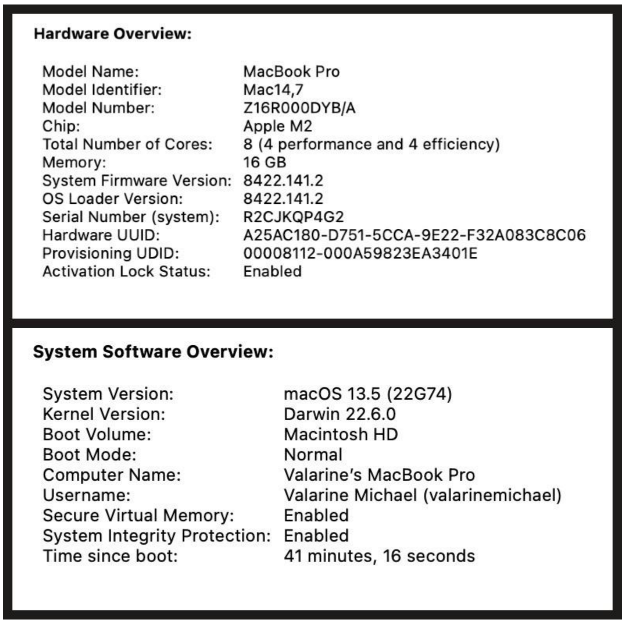

Figure 1: System Configuration

2.1 Softwares & Hardwares

- MS Office 365: The metadata is used in the form of Comma Separated Values (CSV) file.

- Anaconda Navigator: Python version is 3.9.7, Jupyter Notebook version is 6.4.5

3 Packages & Libraries

Importing the required packages and libraries is mandatory before performing data analysis on the data. The list of libraries utilised for this project is displayed in Figure 2.

```
import re
import string
import nltk
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from nltk.stem import PorterStemmer
from nltk.tokenize import word_tokenize
# from genetic_algorithm import GeneticAlgorithm  # Import the genetic algorithm library
import warnings
from sklearn.model_selection import train_test_split
warnings.filterwarnings('ignore')
```

```
import os
import re
import shutil
import string
import tensorflow as tf
import pandas as pd
from tensorflow.keras import layers
from tensorflow.keras import losses
import matplotlib.pyplot as plt
```

```
from tensorflow.keras.layers import Activation, Dense, Embedding, LSTM, SpatialDropout1D, Dropout, Flatten, GRU, Conv1D
from tensorflow.keras.models import Sequential, load_model
from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping, ReduceLROnPlateau
```

```
X_DeepLearning.shape
```

```
(162973, 180)
```

Figure 2: Libraries Used for this Project.

4 Dataset

For this project, a public dataset called Cyberbullying sentiment analysis dataset is used from all social media. The dataset can be accessed from https://github.com/val-elza/Thesis---Genetic-Algorithm-Based-Sentiment-Analysis-for-Cyberbullying-Detection

5 Data Pre-processing

The figure below shows the data pre-processing by feature extraction through TF-IDF (Term Frequency-Inverse Document Frequency) Vectorization process.

```
df.head()
```

	clean_text	category
0	when modi promised “minimum government maximum...	-1.0
1	talk all the nonsense and continue all the dra...	0.0
2	what did just say vote for modi welcome bjp t...	1.0
3	asking his supporters prefix chowkidar their n...	1.0
4	answer who among these the most powerful world...	1.0

```
df.dropna(subset=['category'], inplace=True)
```

```
stopwords_set = set(stopwords.words('english'))
def preprocess_text(text):
        # Check if text is a string
        if isinstance(text, str):
            # Convert text to lowercase
            text = text.lower()

            # Remove URLs
            text = re.sub(r'http\S+|www\S+', '', text)

            # Remove numbers
            text = re.sub(r'\d+', '', text)

            # Remove punctuation
            text = text.translate(str.maketrans('', '', string.punctuation))

            # Tokenization
            tokens = text.split()

            # Remove stopwords

            tokens = [word for word in tokens if word not in stopwords_set]

            # Lemmatization
            lemmatizer = WordNetLemmatizer()
            tokens = [lemmatizer.lemmatize(word) for word in tokens]

            # Join tokens back into a single string
            cleaned_text = ' '.join(tokens)

            return cleaned_text

        # Return empty string if text is not a string
        return ''
```

```
df['clean_text'] = df['clean_text'].apply(preprocess_text)
```

```
df.dropna(subset=['clean_text'], inplace=True)
df.dropna(subset=['category'], inplace=True)
```

```
X = df['clean_text'].values
# y = tf.keras.utils.to_categorical(df['category'], num_classes=len(df['category'].unique()))

y= df['category']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Figure 3: Data Pre-Processing

6 Classification Models

The below classification models were trained and tested on the sentiment analysis task by predicting the sentiment (positive, negative, or neutral) of text data using TF-IDF features. The analysis results, which include the accuracy, classification report, and confusion matrix, are utilised to evaluate each model's performance.

The construction of models is shown in a snippet of code in Figure 4. Six different machine learning models and deep learning was used in creation of this study.

```
epochs = 10
emb_dim = 256
batch_size = 50
model_lstm1 = Sequential()
model_lstm1.add(tf.keras.Input(shape=(X_DeepLearning.shape[1],)))
model_lstm1.add(Embedding(vocabulary_size,emb_dim, input_length=X_DeepLearning.shape[1],) )
model_lstm1.add(SpatialDropout1D(0.8))
model_lstm1.add(Bidirectional(LSTM(300, dropout=0.5, recurrent_dropout=0.5)))
model_lstm1.add(Dropout(0.5))
model_lstm1.add(Flatten())
model_lstm1.add(Dense(64, activation='relu'))
model_lstm1.add(Dropout(0.5))
model_lstm1.add(Dense(3, activation='softmax'))
model_lstm1.compile(optimizer=tf.optimizers.Adam(),loss='categorical_crossentropy', metrics=['acc'])
print(model_lstm1.summary())
```

```
WARNING:absl:At this time, the v2.11+ optimizer `tf.keras.optimizers.Adam` runs slowly on M1/M2 Macs, please use the
legacy Keras optimizer instead, located at `tf.keras.optimizers.legacy.Adam`.
WARNING:absl:There is a known slowdown when using v2.11+ Keras optimizers on M1/M2 Macs. Falling back to the legacy K
eras optimizer, i.e., `tf.keras.optimizers.legacy.Adam`.
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 180, 256)          51200000

 spatial_dropout1d (Spatial  (None, 180, 256)          0
 Dropout1D)

 bidirectional (Bidirection  (None, 600)               1336800
 al)

 dropout (Dropout)           (None, 600)               0

 flatten (Flatten)           (None, 600)               0

 dense (Dense)               (None, 64)                38464

 dropout_1 (Dropout)         (None, 64)                0

 dense_1 (Dense)             (None, 3)                 195

=================================================================
Total params: 52575459 (200.56 MB)
Trainable params: 52575459 (200.56 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
None
```

```
checkpoint_callback = ModelCheckpoint(filepath="lastm-1-layer-best_model.h5", save_best_only=True, monitor="val_acc", m

early_stopping_callback = EarlyStopping(monitor="val_acc", mode="max", patience=10, verbose=1, restore_best_weights=Tru

reduce_lr_callback = ReduceLROnPlateau(monitor="val_loss", factor=0.1, patience=5, verbose=1, mode="min", min_delta=0.0

callbacks=[checkpoint_callback, early_stopping_callback, reduce_lr_callback]
```

```
history_lstm1 = model_lstm1.fit(XX_train, y_train, epochs = epochs, batch_size = 250, validation_data=(XX_test, y_test)
```

```
Epoch 1/10
489/489 [==============================] - ETA: 0s - loss: 0.7182 - acc: 0.6985
Epoch 1: val_acc improved from -inf to 0.86501, saving model to lastm-1-layer-best_model.h5
489/489 [==============================] - 985s 2s/step - loss: 0.7182 - acc: 0.6985 - val_loss: 0.4002 - val_acc: 0.
8650 - lr: 0.0010
Epoch 2/10
489/489 [==============================] - ETA: 0s - loss: 0.4068 - acc: 0.8683
Epoch 2: val_acc improved from 0.86501 to 0.89608, saving model to lastm-1-layer-best_model.h5
489/489 [==============================] - 1025s 2s/step - loss: 0.4068 - acc: 0.8683 - val_loss: 0.3374 - val_acc:
0.8961 - lr: 0.0010
Epoch 3/10
489/489 [==============================] - ETA: 0s - loss: 0.3418 - acc: 0.8938
```

```python
from sklearn.feature_extraction.text import TfidfVectorizer
```

```python
from sklearn.linear_model import SGDClassifier, LogisticRegression
```

```python
X = df['clean_text'].values
# y = tf.keras.utils.to_categorical(df['category'], num_classes=len(df['category'].unique()))

y= df['category']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```python
vectorizer = TfidfVectorizer()
X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf = vectorizer.transform(X_test)
```

```python
logistic_regression = LogisticRegression()
logistic_regression.fit(X_train_tfidf, y_train)
```

```
▾ LogisticRegression
LogisticRegression()
```

```python
from sklearn.metrics import classification_report, accuracy_score, confusion_matrix
```

```python
accuracy_score(y_test, _pred)
```

```
0.879306642123025
```

```python
import numpy as np
```

```python
y_train.to_list()
```

```
 0.0,
 0.0,
 1.0,
 0.0,
 -1.0,
 -1.0,
 0.0,
 1.0,
 0.0,
 1.0,
 1.0,
 -1.0,
 0.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
```

```python
from sklearn.svm import SVC
```

```python
np.any(np.isnan(y_train))
```

```
False
```

```python
svm = SVC(decision_function_shape='ovo')
svm.fit(X_train_tfidf, y_train.to_list())
```

```
▾ SVC
SVC(decision_function_shape='ovo')
```

```python
svc_prd = svm.predict(X_test_tfidf)
```

```python
accuracy_score(y_test, svc_prd)
```

```
0.8833563429973922
```

Figure 4: Deep and Machine Learning Models

7 Implementation of Code

- Download Cyberbullying dataset from GitHub link provide in Section 4.
- Download “Thesis_Project.zip”, unzip it and create a folder called Sem-2 and create a subfolder called Thesis and save the dataset as “Twitter_Data.csv”.
- Unzip the downloaded dataset into the newly created Thesis folder.
- Run the script and wait for the models to get trained. Finally, the machine learning model is also completed.
- You will then receive an output.