National College of Ireland

Configuration Manual

MSc Research Project
MSc Data Analytics

Rishabh Singh Chauhan
Student ID:X21107939

School of Computing
National College of Ireland

Supervisor: Vladimir Milosavljevic

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Rishabh Singh Chauhan		
Student ID:	X21107939		
Programme:	MSc Data Analytics	**Year:**	2022-2023
Module:	Msc Research Project		
Lecturer:	Vladimir Milosavljevic		
Submission Due Date:	15 December 2022		
Project Title:	Sentiment Analysis of Customer Reviews Using Deep Learning		
Word Count:	**492**	**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: Rishabh Singh Chauhan

Date: 15 December 2022

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

Rishabh Singh Chauhan
X21107939

1. Introduction

This configuration manual provides all the necessary information and procedures that are needed to reproduce the result of the project Sentiment analysis of customer reviews using deep learning.

2. System Requirements

Hardware Configurations needed to run the code.

Device name	RISHABH
Processor	11th Gen Intel(R) Core(TM) i5-1155G7 @ 2.50GHz 2.50 GHz
Installed RAM	16.0 GB (15.8 GB usable)
Device ID	E1D92E41-7DB2-4988-8C08-964DB45D817A
Product ID	00342-42598-29368-AAOEM
System type	64-bit operating system, x64-based processor
Pen and touch	No pen or touch input is available for this display

Software Configurations:-
1. Anaconda Jupyter Notebook
2. Python 3.7
3. Anaconda Navigator
4. MS Office

3. Data Collection

The dataset can be downloaded from Kaggle through this link
https://www.kaggle.com/datasets/therealsampat/google-play-apps-reviews
The above dataset is in the form of a CSV file.

4. Libraries required in Anaconda Runtime Environment

```
import re
import matplotlib.pyplot as plt
import string
from nltk.corpus import stopwords
import nltk
from nltk.corpus import wordnet
from nltk.stem import WordNetLemmatizer
from nltk.tokenize.treebank import TreebankWordDetokenizer
from collections import Counter
from wordcloud import WordCloud
from nltk.corpus import stopwords
import nltk
from gensim.utils import simple_preprocess
from nltk.corpus import stopwords
import gensim
from sklearn.model_selection import train_test_split
# import spacy
import pickle
import warnings
warnings.filterwarnings('ignore')
import seaborn as sns
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
import tensorflow as tf
import keras
import numpy as np
import pandas as pd
```

```
# Import necessary libraries
import numpy as np
import pandas as pd
import seaborn as sns
from pylab import rcParams
import matplotlib.pyplot as plt
from matplotlib import rc
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, classification_report
from collections import defaultdict
from textwrap import wrap

# Torch ML libraries
import transformers
from transformers import BertModel, BertTokenizer, AdamW, get_linear_schedule_with_warmup
import torch
from torch import nn, optim
from torch.utils.data import Dataset, DataLoader
```

5. Data Preparation

In the dataset, there are 12 columns but only two columns “content” and “score” are used in the research project therefore we have to extract these column and put into our data frame.

```
df = df[['content','score']]
df.head()
```

	content	score
0	I cannot open the app anymore	1
1	I have been begging for a refund from this app...	1
2	Very costly for the premium version (approx In...	1
3	Used to keep me organized, but all the 2020 UP...	1
4	Dan Birthday Oct 28	1

Score is labelled into “Positive” , “Neutral” and “Negative”

```
def to_sentiment(rating):

    rating = int(rating)

    # Convert to class
    if rating <= 2:
        return 0
    elif rating == 3:
        return 1
    else:
        return 2

# Apply to the dataset
df['sentiment'] = df.score.apply(to_sentiment)
```

6. Train_Test Split

Train-Test Split

```
9]: X_train, X_test, y_train, y_test = train_test_split(tweets, labels , random_state=0)
    print (len(X_train),len(X_test),len(y_train),len(y_test))
```

```
9371 3124 9371 3124
```

Dataset is divided into 75% train data and 25% test data

7. Model Implementation

First Import Keras Library

Importing Keras Library

```
from keras.models import Sequential
from keras import layers
from keras.optimizers import RMSprop,Adam
from keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pa
from keras import regularizers
from keras import backend as K
from keras.callbacks import ModelCheckpoint
max_words = 5000
max_len = 200

tokenizer = Tokenizer(num_words=max_words)
tokenizer.fit_on_texts(data)
sequences = tokenizer.texts_to_sequences(data)
tweets = pad_sequences(sequences, maxlen=max_len)
print(tweets)
```

Model 1- Long Short-Term Memory

LSTM Model

```
model1 = Sequential()
model1.add(layers.Embedding(max_words, 20))
model1.add(layers.LSTM(15,dropout=0.5))
model1.add(layers.Dense(3,activation='softmax'))


model1.compile(optimizer='adam',loss='categorical_crossentropy', metrics=['accuracy'])
#Implementing model checkpoins to save the best metric and do not lose it on training.
checkpoint1 = ModelCheckpoint("best_model1.hdf5", monitor='val_accuracy', verbose=1,save_best_only=True, mode='auto', period=1,sa
history = model1.fit(X_train, y_train, epochs=20,validation_data=(X_test, y_test),callbacks=[checkpoint1])
```

Model 2 - Bidirectional Recurrent Neural Networks

Model 2 Bidirectional LSTM

```
model2 = Sequential()
model2.add(layers.Embedding(max_words, 40, input_length=max_len))
model2.add(layers.Bidirectional(layers.LSTM(20,dropout=0.6)))
model2.add(layers.Dense(3,activation='softmax'))
model2.compile(optimizer='rmsprop',loss='categorical_crossentropy', metrics=['accuracy'])
#Implementing model checkpoins to save the best metric and do not lose it on training.
checkpoint2 = ModelCheckpoint("best_model2.hdf5", monitor='val_accuracy', verbose=1,save_best_only=True, mode='auto', period=1,sa
history = model2.fit(X_train, y_train, epochs=20,validation_data=(X_test, y_test),callbacks=[checkpoint2])
```

Model 3- Convolutional 1d

Model 3 Convolutional 1D

```
from keras import regularizers
model3 = Sequential()
model3.add(layers.Embedding(max_words, 40, input_length=max_len))
model3.add(layers.Conv1D(20, 6, activation='relu',kernel_regularizer=regularizers.l1_l2(l1=2e-3, l2=2e-3),bias_regularizer=regula
model3.add(layers.MaxPooling1D(5))
model3.add(layers.Conv1D(20, 6, activation='relu',kernel_regularizer=regularizers.l1_l2(l1=2e-3, l2=2e-3),bias_regularizer=regula
model3.add(layers.GlobalMaxPooling1D())
model3.add(layers.Dense(3,activation='softmax'))
model3.compile(optimizer='rmsprop',loss='categorical_crossentropy',metrics=['acc'])
checkpoint3 = ModelCheckpoint("best_model3.hdf5", monitor='val_accuracy', verbose=1,save_best_only=True, mode='auto', period=1,sa
history = model3.fit(X_train, y_train, epochs=20,validation_data=(X_test, y_test),callbacks=[checkpoint3])
```

```
WARNING:tensorflow:`period` argument is deprecated. Please use `save_freq` to specify the frequency in number of batches seen.
```

Model 4- Bidirectional Encoder Representations from Transformer

First, install the transformers through pip install transformer. The BERT model comes into two sections bert_preprocess and bert_encoder install both the models .

Now lets import BERT model and get embeding vectors for few sample statements

```
bert_preprocess = hub.KerasLayer("https://tfhub.dev/tensorflow/bert_en_uncased_preprocess/3")
bert_encoder = hub.KerasLayer("https://tfhub.dev/tensorflow/bert_en_uncased_L-12_H-768_A-12/4")
```

8 Model Evaluation

To evaluate the performance of the sklearn. metrics are used

```
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score, precision_score, recall_score
print(confusion_matrix(y_test.argmax(axis=1), np.around(pred, decimals=0).argmax(axis=1)))
print(classification_report(y_test.argmax(axis=1), np.around(pred, decimals=0).argmax(axis=1)))
print(accuracy_score(y_test.argmax(axis=1), np.around(pred, decimals=0).argmax(axis=1)))
```

```
[[1078    0  128]
 [ 346    1  173]
 [ 342    0 1056]]
              precision    recall  f1-score   support

           0       0.61      0.89      0.73      1206
           1       1.00      0.00      0.00       520
           2       0.78      0.76      0.77      1398

    accuracy                           0.68      3124
   macro avg       0.80      0.55      0.50      3124
weighted avg       0.75      0.68      0.62      3124

0.6834186939820742
```