Configuration Manual

MSc Research Project
Data Analytics

Annjoys Robert
StudentID: 22137459

School of Computing
National College of Ireland

Supervisor: Vladimir Milosavljevic

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Annjoys Robert
Student ID:	22137459
Programme:	Data Analytics
Year:	2023
Module:	MSc Research Project
Supervisor:	Vladimir Milosavljevic
Submission Due Date:	14/12/2023
Project Title:	Configuration Manual
Word Count:	590
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:	Annjoys Robert
Date:	14th December 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

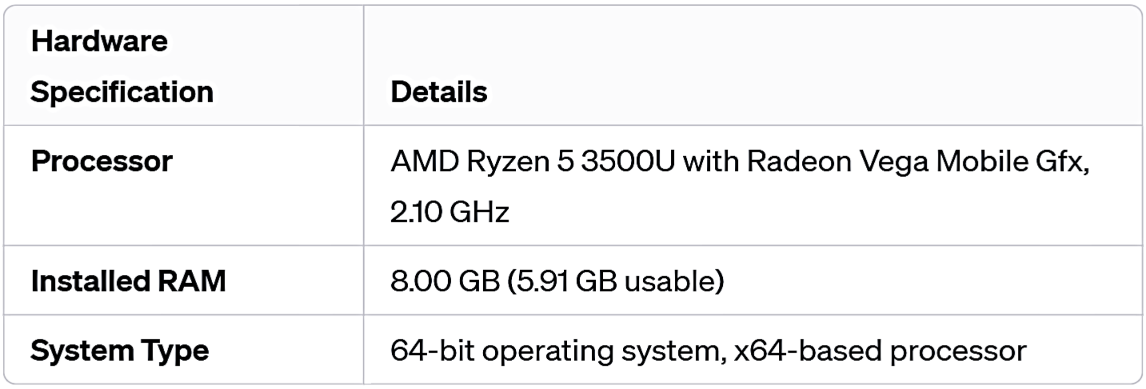
Annjoys Robert
22137459

1 Introduction

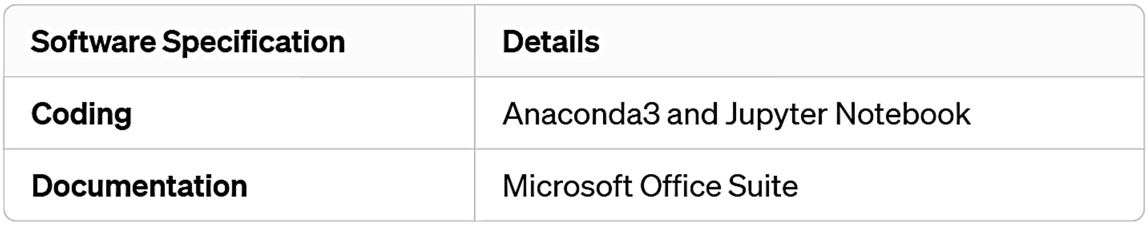
This research performs sentiment analysis on product reviews using machine learning models such as CNN, RNN, LSTM, and BERT. This manual details the setup and execution of the current research project's scripts. It provides guidance on running the code smoothly, including recommended hardware and software versions. Following these instructions precisely will enable the replication of the project's results.

2 System Configurations

Hardware and Software Configuration



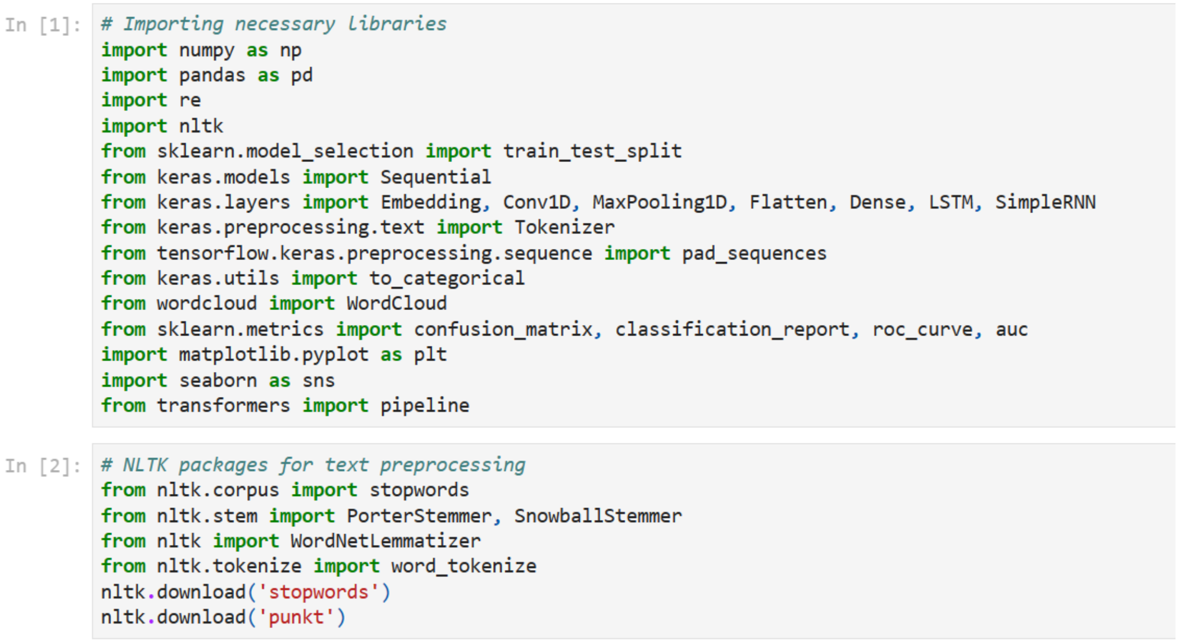
Hardware Specification	Details
Processor	AMD Ryzen 5 3500U with Radeon Vega Mobile Gfx, 2.10 GHz
Installed RAM	8.00 GB (5.91 GB usable)
System Type	64-bit operating system, x64-based processor



Software Specification	Details
Coding	Anaconda3 and Jupyter Notebook
Documentation	Microsoft Office Suite

3 Data Preparation and Text Preprocessing

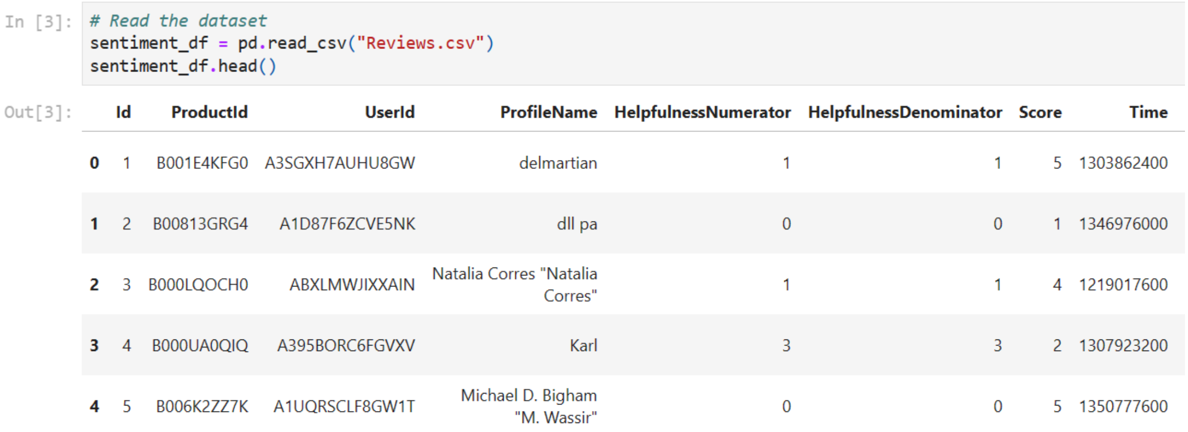
First Importing of necessary libraries and packages was performed as shown in Figure 1. Later loading the data using Pandas done as shown in Figure 2, Cleaning data by removing duplicates, handling null values, and standardizing text format was performed as shown in Figure 3 and 4. Removing HTML tags, numbers, special characters and utiliz NLP techniques like tokenization, stop-word removal, stemming, and lemmatization.



```
In [1]: # Importing necessary libraries
        import numpy as np
        import pandas as pd
        import re
        import nltk
        from sklearn.model_selection import train_test_split
        from keras.models import Sequential
        from keras.layers import Embedding, Conv1D, MaxPooling1D, Flatten, Dense, LSTM, SimpleRNN
        from keras.preprocessing.text import Tokenizer
        from tensorflow.keras.preprocessing.sequence import pad_sequences
        from keras.utils import to_categorical
        from wordcloud import WordCloud
        from sklearn.metrics import confusion_matrix, classification_report, roc_curve, auc
        import matplotlib.pyplot as plt
        import seaborn as sns
        from transformers import pipeline
```

```
In [2]: # NLTK packages for text preprocessing
        from nltk.corpus import stopwords
        from nltk.stem import PorterStemmer, SnowballStemmer
        from nltk import WordNetLemmatizer
        from nltk.tokenize import word_tokenize
        nltk.download('stopwords')
        nltk.download('punkt')
```

Figure 1: Importing necessary libraries and packages.

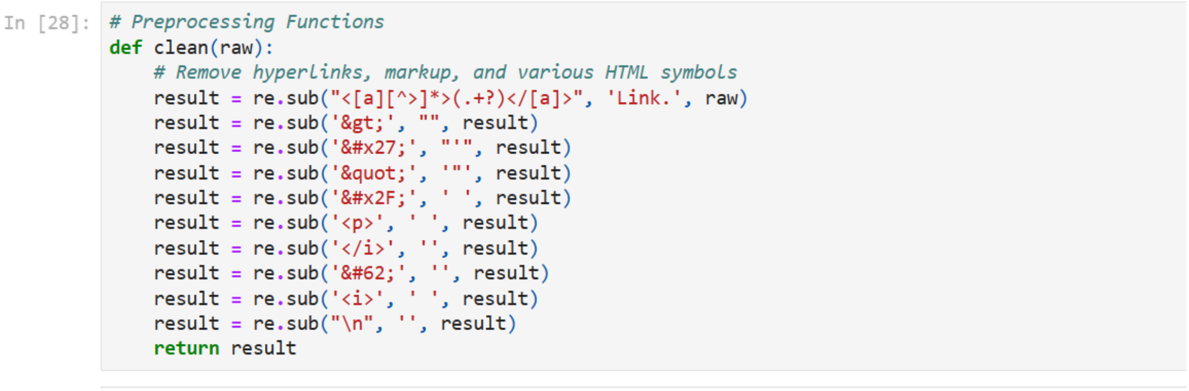


```
In [3]: # Read the dataset
        sentiment_df = pd.read_csv("Reviews.csv")
        sentiment_df.head()
```

Out[3]:

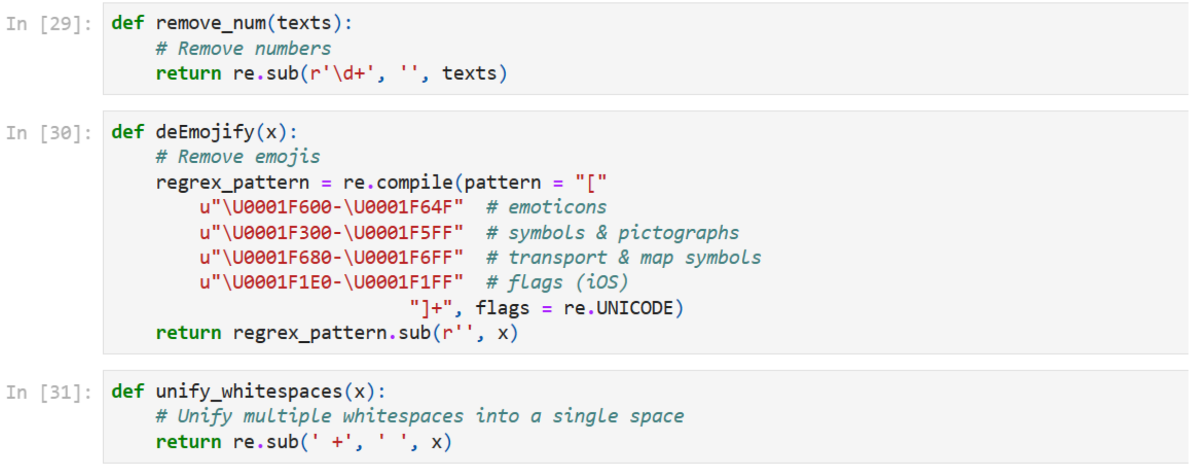
	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303862400
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	1346976000
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	4	1219017600
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	2	1307923200
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	5	1350777600

Figure 2: Loading the dataset.



```
In [28]: # Preprocessing Functions
         def clean(raw):
             # Remove hyperlinks, markup, and various HTML symbols
             result = re.sub("<[a][^>]*>(.+?)</[a]>", 'Link.', raw)
             result = re.sub('&gt;', "", result)
             result = re.sub('&#x27;', "'", result)
             result = re.sub('&quot;', '"', result)
             result = re.sub('&#x2F;', ' ', result)
             result = re.sub('<p>', ' ', result)
             result = re.sub('</i>', '', result)
             result = re.sub('&#62;', '', result)
             result = re.sub('<i>', ' ', result)
             result = re.sub("\n", '', result)
             return result
```

Figure 3: Cleaning the dataset.

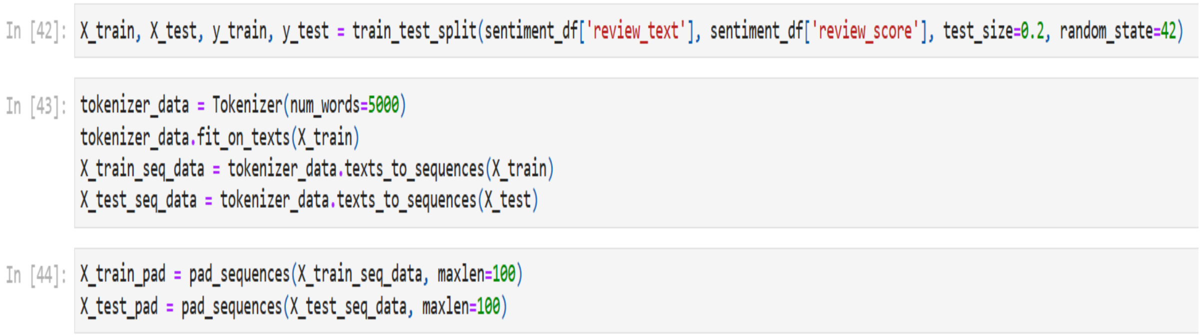


```
In [29]: def remove_num(texts):
             # Remove numbers
             return re.sub(r'\d+', '', texts)

In [30]: def deEmojify(x):
             # Remove emojis
             regrex_pattern = re.compile(pattern = "["
                 u"\U0001F600-\U0001F64F"  # emoticons
                 u"\U0001F300-\U0001F5FF"  # symbols & pictographs
                 u"\U0001F680-\U0001F6FF"  # transport & map symbols
                 u"\U0001F1E0-\U0001F1FF"  # flags (iOS)
                                    "]+", flags = re.UNICODE)
             return regrex_pattern.sub(r'', x)

In [31]: def unify_whitespaces(x):
             # Unify multiple whitespaces into a single space
             return re.sub(' +', ' ', x)
```

Figure 4: Other data preprocesssing fuctions.

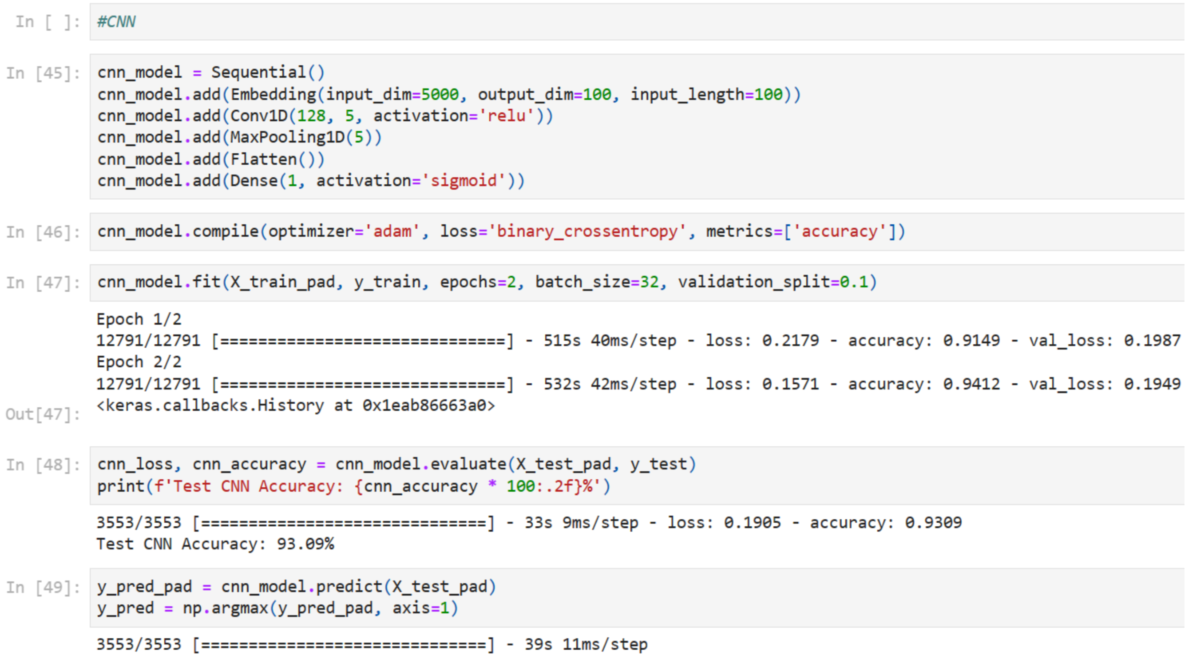
4 Model Configuration and Training

Building and training CNN, RNN, LSTM, BERT models using Keras. Detailed guide on setting hyperparameters, layers, and training process was performed under model configuration and training as shown in figure 5 and 6.



```
In [42]: X_train, X_test, y_train, y_test = train_test_split(sentiment_df['review_text'], sentiment_df['review_score'], test_size=0.2, random_state=42)

In [43]: tokenizer_data = Tokenizer(num_words=5000)
         tokenizer_data.fit_on_texts(X_train)
         X_train_seq_data = tokenizer_data.texts_to_sequences(X_train)
         X_test_seq_data = tokenizer_data.texts_to_sequences(X_test)

In [44]: X_train_pad = pad_sequences(X_train_seq_data, maxlen=100)
         X_test_pad = pad_sequences(X_test_seq_data, maxlen=100)
```

Figure 5: Model training.



```
In [ ]: #CNN

In [45]: cnn_model = Sequential()
         cnn_model.add(Embedding(input_dim=5000, output_dim=100, input_length=100))
         cnn_model.add(Conv1D(128, 5, activation='relu'))
         cnn_model.add(MaxPooling1D(5))
         cnn_model.add(Flatten())
         cnn_model.add(Dense(1, activation='sigmoid'))

In [46]: cnn_model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

In [47]: cnn_model.fit(X_train_pad, y_train, epochs=2, batch_size=32, validation_split=0.1)

         Epoch 1/2
         12791/12791 [==============================] - 515s 40ms/step - loss: 0.2179 - accuracy: 0.9149 - val_loss: 0.1987
         Epoch 2/2
         12791/12791 [==============================] - 532s 42ms/step - loss: 0.1571 - accuracy: 0.9412 - val_loss: 0.1949
Out[47]: <keras.callbacks.History at 0x1eab86663a0>

In [48]: cnn_loss, cnn_accuracy = cnn_model.evaluate(X_test_pad, y_test)
         print(f'Test CNN Accuracy: {cnn_accuracy * 100:.2f}%')

         3553/3553 [==============================] - 33s 9ms/step - loss: 0.1905 - accuracy: 0.9309
         Test CNN Accuracy: 93.09%

In [49]: y_pred_pad = cnn_model.predict(X_test_pad)
         y_pred = np.argmax(y_pred_pad, axis=1)

         3553/3553 [==============================] - 39s 11ms/step
```

Figure 6: Model Implementation (CNN).

5 Evaluation and Visualization

Evaluating the models using accuracy, confusion matrix, ROC curve was done and visualizations like word clouds and distribution plots were created as shown in figure 7 and 8.

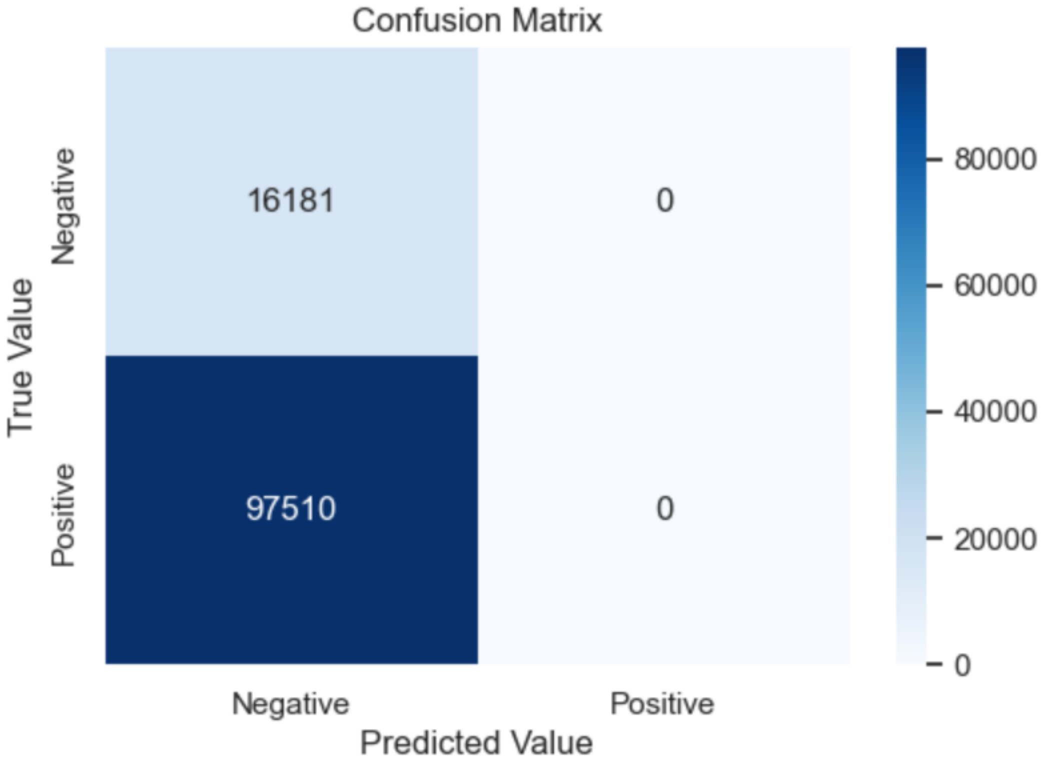


Figure 7: Confusion Matrix (CNN)

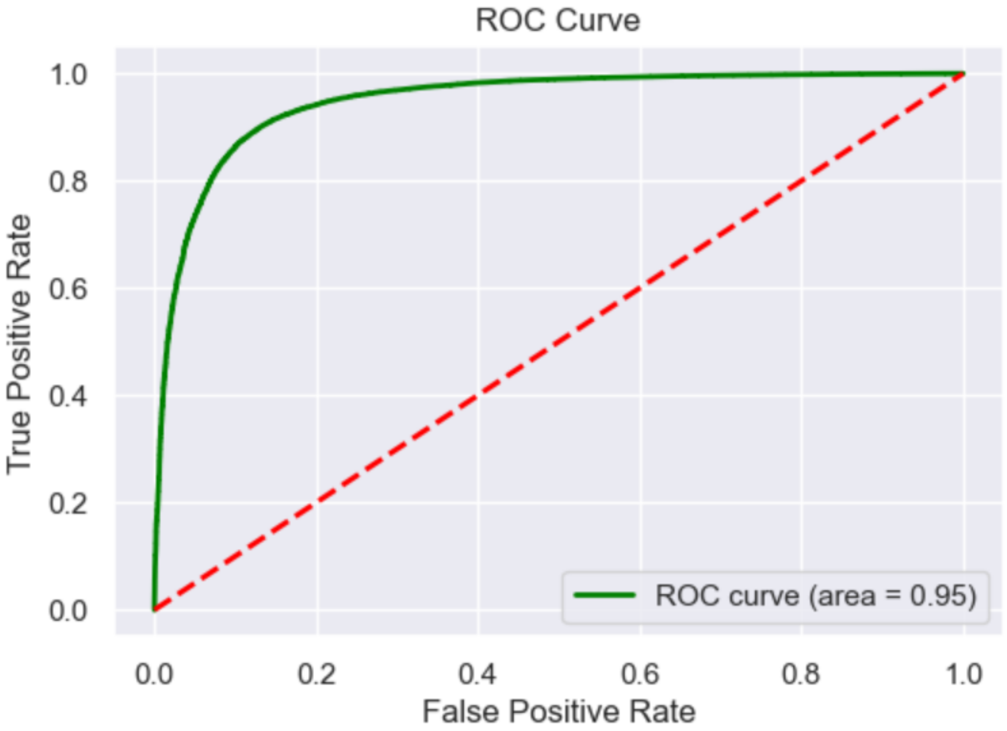


Figure 8: ROC curve (CNN)

References

J. S. Vimali and S. Murugan, "A Text Based Sentiment Analysis Model using Bi-directional LSTM Networks," 2021 6th International Conference on Communication and Electronics Systems (ICCES), Coimbatre, India, 2021, pp. 1652-1658, doi: 10.1109/ICCES51350.2021.9489129.

M. R. Bhuiyan, M. H. Mahedi, N. Hossain, Z. N. Tumpa and S. A. Hossain, "An Attention Based Approach for Sentiment Analysis of Food Review Dataset," 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Kharagpur, India, 2020, pp. 1-6, doi: 10.1109/ICCCNT49239.2020.9225637.