Mini Project 2

Contents

By:

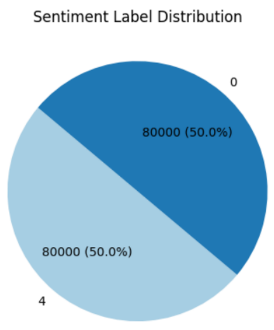
Mohammadreza Akhtari

Introduction

In this project, we are going to analyse the tweets that have been posted on Twitter to determine whether they are positive or negative. In other words, the task is to analyse text or natural language processing (NLP). We are provided with the comments' database which could be easily read for further investigation.

Data processing

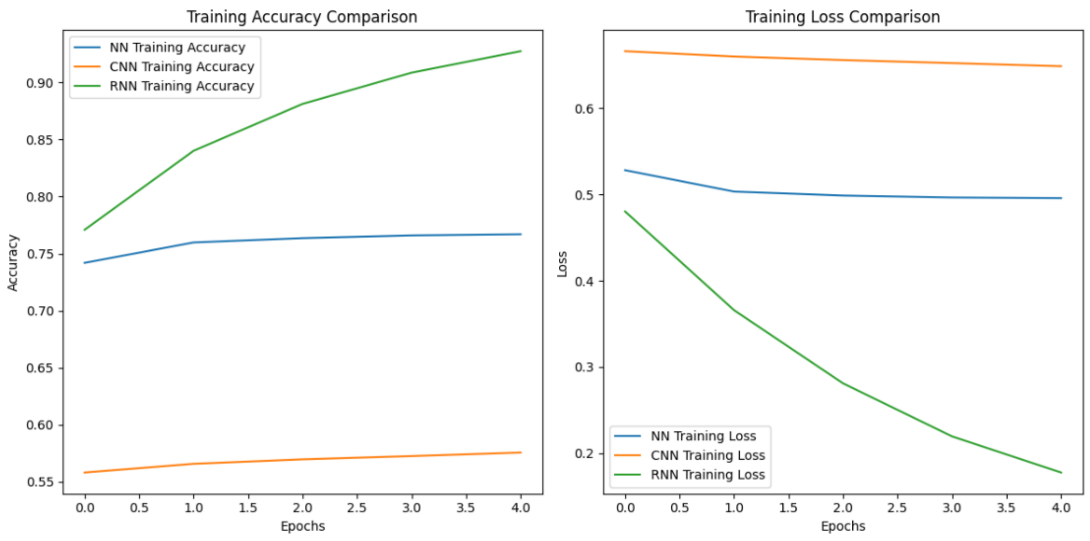
To start processing the test, we need to do pre-processing or pre-treat the data. To achieve this, we check for empty or invalid input such as empty rows, characters that make no sense in language processing such as mentioning others or using symbols, etc. which are removed from the dataset to have a cleaner data set. Data are balanced meaning that positive and negative database has the same amount of records as below. Each class has 80,000 data. Null and missing values have been also checked after removing symbols and numbers to have better and more useful database.



Modelling

To perform our model, it is necessary to have tokenization meaning that words are extracted and then later, they will be assigned some numbers to make them numeric and easier to manipulate. The satisfaction criteria have been also transformed to 0 and 1 instead of 0 and 4 for better data management. Data are divided into 20-80 % groups for training and testing. The maximum length of comments after cleaning is 169. Three models have been tested namely neural network, convolutional neural network and recurrent neural network to select the appropriate model.

Three models are compared against each other in the following graph.



As can be seen, RNN or recurrent neural network has the highest accuracy followed by convolutional neural network. Also, RNN has the lowest loss while CNN occupies the second place.

The models have been deployed kind of similarly to the lecture to achieve the best results. Taking advantage of Glove, the pre-trained code is embedded in our modelling to achieve numeric values of the tokenised sentences and to make the relationship between words.

Conclusions

This kind of training is a really time-consuming and computationally intensive problem. It takes a lot of time to run the codes and train the models. However, the results are promising without using too much-complicated models. RNN is obviously the best choice here with the lowest loss among two others. The accuracy of the model may be increased by adding more layers and manipulating parameters.

RNN typically works better, especially with sequential data having some degree of flexibility or kind of dynamic behaviour of the model. However, its computational intensive requirement should be considered to select the best model which fits the appropriate