DEEP TWEETS REPORT

JUNE 12, 2019

BIHANI MOHAMED AMINE

Presentation

In this task report we will treat a tweet classification problem. We will build a machine learning model that can classify tweets either Politics or Sports.

The data used in this task are the training data and the testing data provided by Kaggle's platform for tweets classification.

We will also make usage of a Kaggle text file for our words indexing as values.

I. Extracting and preparing the data

1. Importing Libraries.

We import the libraries that we will use for cleaning the dataset. We will also import more libraries later in case we need them for a new approach or calculations within the model.

2. Importing the dataset.

Next up we import our dataset (training and testing) in csv format, using pandas of course.

3. Cleaning data.

In order to prepare the data, we must clean it thoroughly because a good cleaned data ensures a better prediction. So, we have to remove commas, links, punctuation, stopwords, tags (@) and special characters. Then we tokenize the words in order to have a list that will ease the training process as it will give certain weights to words in order to reach the general value that will give the prediction on the whole list.

4. Preparing the embedding layer

We use keras preprocessing module to create a word to index dictionary, as we will use this to make an embedding matrix using GloVe embeddings to create our feature matrix. (this is a text file that we can download separately from Kaggle) We

will create a dictionary that will contain words as keys and their corresponding embedding list as values. This will be the basis of our training method.

5. **Setting up the values**

We change the labels sports and Politics to single values 0 and 1 respectively. That is to change the words values so that it is affected depending on their relation to sports or politics. Therefore, our prediction result will be a number between 0 and 1, and if it is above 0.5 then it is more bound to Politics and vice versa.

II. Model training

I chose to approach this problem using CNN (convolutional neural networks) as I had just learned about their usage and decided to test them. Even if it is not the best fit for our data, it was a new experience for me. The basis of this model is using 1 convolutional layer and 1 pooling layer. we create a one-dimensional convolutional layer with 128 features, or kernels. The kernel size is 5 and the activation function used is sigmoid. Next, we add a global max pooling layer to reduce feature size (in this case 500). Finally, we add a dense layer with sigmoid activation.

III. Process and Tools

1-Libraries :

Pandas: library of data manipulation and analysis.
Ntlk: for stopwords.
Keras: for preprocessing, embedding layers.
Sklearn: splitting data for training.
Seaborn: Visualization
Numpy: Characters
Matplotlib.pyplot: Visualization

2-Process of implementation :

-Importing Libraries

-Importing the training data and testing data

-Saving the column of TweetText of the testing data.

-Cleaning the training data

-Setting up the labels

-Tokenizing

-Preparing embedding layers and matrix

-Preparing and running the model

-Visualizing the performance

-Submitting our results to the csv file named submission.csv

3-Tools used

Python 3.0 on Google colab IDE. It gives access to high speed download and upload of files and has many useful features in terms of importing and working on data quickly.

IV. Conclusion

The testing gave me an accuracy of 0.92 on the training dataset.
The first submission on Kaggle landed however only 0.841, and my second one with a slightly improved cleaning process gave 0.850. So only a slight improvement. There is still room for improvement because I wanted to reach a score above 0.9.