Name: Naresh Kumar J

Email address: nareshkumar1742@gmail.com

Contact number: 9489030727

Anydesk address:

Years of Work Experience: **Nil**

Date: **29/10/22**

Self Case Study -2: Tweet Sentiment Extraction

"After you have completed the document, please submit it in the classroom in the pdf format."

Please check this video before you get started:
https://www.youtube.com/watch?time_continue=1&v=LBGU1_JO3kg

Overview

*** Write an overview of the case study that you are working on. ***(MINIMUM 200 words)*** ***

1. With all of the tweets circulating every second it is hard to tell whether the sentiment behind a specific tweet will impact a company, or a person's, brand for being viral (positive), or devastate profit because it strikes a negative tone. Capturing sentiment in language is important in these times where decisions and reactions are created and updated in seconds. But, which words actually lead to the sentiment description? In this competition we will need to pick out the part of the tweet (word or phrase) that reflects the sentiment. The main objective of this study is to figure out which phrase or word determines the sentiment of the tweet.

Research-Papers/Solutions/Architectures/Kernels

*** Mention the urls of existing research-papers/solutions/kernels on your problem statement and in your own words write a detailed summary for each one of them. If needed you can include images or explain with your own diagrams. ***

1. https://medium.com/analytics-vidhya/tweet-sentiment-extraction-e5d242ff93c2

   In this blog the author tried base model approach by directly giving the text input and the sentiment input and calculate the start index and the end index using the LSTM and Bidirectional LSTM. The base model showed result of 0.5 Jaccard score with base LSTM and 0.6 Jaccard score with bidirectional LSTM.

2. https://medium.com/@ashish.sarathe/tweet-sentiment-extraction-dc9b4660dd1b

   In this blog the author tried with the same above mentioned way. But here there were some advancement in the preprocessing and little changes in the output (i.e instead of start and end index the output consist of binary format with the length of maximum words in a sentence. 1 represents that the particular positioned word is a selected text.) thus the base model score was higher than the previous one. The Jaccard score for the base model with LSTM was 0.58 and with bidirectional LSTM he was able to achieve 0.60.
3. https://towardsdatascience.com/train-ner-with-custom-training-data-using-spacy-525ce748fab7

   In this blog the author explains how we can use spacy library to train custom Named Entity Recognition for our own dataset.

First Cut Approach

*** Explain in steps about how you want to approach this problem and the initial experiments that you want to do. ***(MINIMUM 200 words)*** ***

1. In the above both the references the loss function that was used was mean square error in the first reference and binary cross entropy for the second reference. Here the main objective is to maximize the jaccard score. So to do that I have decided to use Dice loss as loss and iou score as metric to have a better control of the model.
   https://github.com/qubvel/segmentation_models

2. From the third reference I tried building my own custom NER model to see how the well does this model gives us the result.

3. Finally used BERT for modeling. In the above first and second references we can see that the authors used the output layer (i.e the start logits and the end logits of the bert) and on top of that they built their architecture (i.e either used dense layer or dropout layer). Here instead of using the final layer of the bert (i.e layer that gives start logit and end logit) I

used the output of the previous layer to the final layer which is a 3 dimensional output that gives vector embedding from each input word. This layer gives vector for each word which can be considered as word embedding. Since the weights in the BERT are pretrained these word embedding will be much useful and on top of that we can define our own architecture which will yield better results. Here I have defined convolutional 1D layer on top of the BERT embedding layer. Also the BERT layer trains along with the self-defined layers.

Notes when you build your final notebook:

1. You should not train any model either it can be a ML model or DL model or Countvectorizer or even simple StandardScalar
2. You should not read train data files
3. The function1 takes only one argument "X" (a single data points i.e 1*d feature) and the inside the function you will preprocess data point similar to the process you did while you featurize your train data
    a. Ex: consider you are doing taxi demand prediction case study (problem definition: given a time and location predict the number of pickups that can happen)
    b. so in your final notebook, you need to pass only those two values
    c. def final(X):

preprocess data i.e data cleaning, filling missing values etc

compute features based on this X

use pre trained model

return predicted outputs

final([time, location])

    d. in the instructions, we have mentioned two functions one with original values and one without it
    e. final([time, location])   # in this function you need to return the predictions, no need to compute the metric

    f. final(set of [time, location] values, corresponding Y values) # when you pass the Y values, we can compute the error metric(Y, y_predict)

4. After you have preprocessed the data point you will featurize it, with the help of trained vectorizers or methods you have followed for your train data

5. Assume this function is like you are productionizing the best model you have built, you need to measure the time for predicting and report the time. Make sure you keep the time as low as possible

6. Check this live session: https://www.appliedaicourse.com/lecture/11/applied-machine-learning-online-course/4148/hands-on-live-session-deploy-an-ml-model-using-apis-on-aws/5/module-5-feature-engineering-productionization-and-deployment-of-ml-models