Final Report

Saturday, December 12, 2020 12:31 AM

1) An overview of the function of the code (i.e., what it does and what it can be used for).

- As part of this project, I am predicting if a given text (tweet) be SARCASM and NOT_SARCASM based on the response text. This can be extend to other text based application like sentiment analysis where we can find if a text is sarcasam or not and based on that mark the text as positive or negative sen timent or retrain the model for sentiment analysis. If we have to repurpose the code, we do have to train it on the text data so that the model learn the underlying details for **better prediction**.

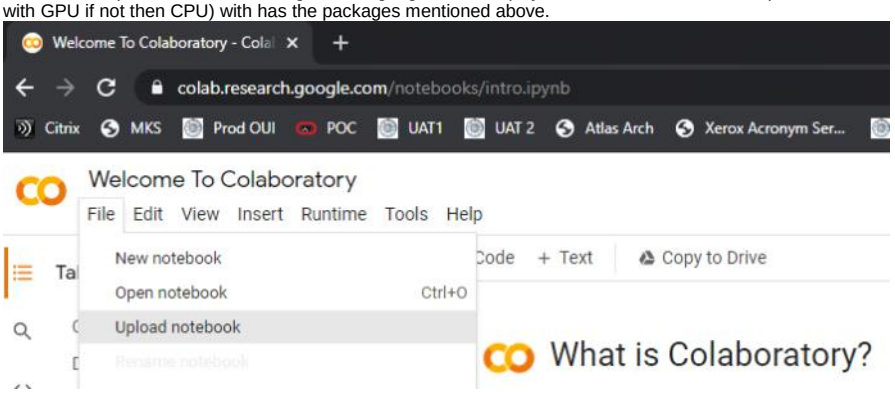
2) Documentation of how the software is implemented with sufficient detail so that others can have a basic understanding of your code for future extension or any further improvement.

- The code is implemented in python and using various library like Pandas, Numpy, transformer, sklearn. Below is the snapshot o f the versions of different packages

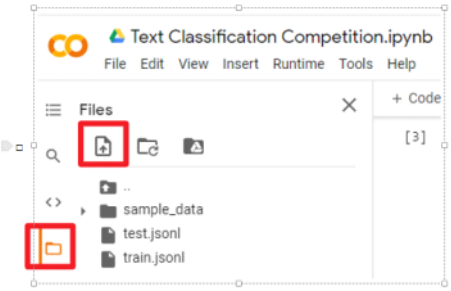
```
Pandas Version 1.1.5
Numpy Version 1.18.5
torch Version 1.7.0+cu101
transformers Version transformers-4.0.1
sklearn Version 0.22.2.post1
matplotlib Version 3.2.2
seaborn Version 0.11.0
Currently selected TF version: 2.x
Available versions:
* 1.x
* 2.x
Python 3.6.9
```

- I have trained the model on the Google Colab using the GPU option. Without the GPU the training was taking 15 hours - 20 hours. With GPU I was able to trained the model in 5 min -10 mins. I have added the comments on the code file for better readability and explaining the steps. Below are the high level steps.
  - About Project
  - About data**:**
  - Importing libraries and loading data
  - Pre Processing of text data :To remove the noise from the text data.
    - Encoding the data to prepend and append the sentence with CLS and SEP token which is needed by BERT
    - padding data : To keep the fixed length for each row of the text data in train and test.
    - creating attention mask : Creating a attention mask for train and test data which is needed for the model
    - Converting data to torch tensor
    - Creating dataloader
  - Loading BertForSequenceClassification model : Loading pretrain BERT model.
  - Creating Scheduler : Needed for BERT
  - Training and validation of the model : Where I am training the train data and doing the testing on validation test
  - Prediction on testing data : Predicting the label for the test data.
  - Generating output file
  - References

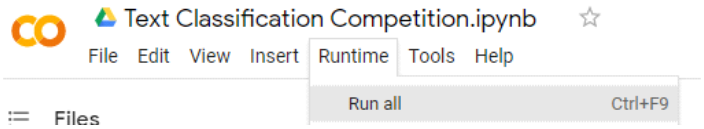
3) Documentation of the usage of the software including either documentation of usages of APIs or detailed instructions on how to install and run a software, whichever is applicable.

- In order to run the code, we need an environment with the above packages. I would recommend to run it on Google Colab with GP U by following the below steps.
  - Go to 'https://colab.research.google.com/notebooks/intro.ipynb'
  - Click on file, upload notebook and sign in with google account (if you don't have an account, if possible create one. If it's not possible, please run it on any machine (preferred with GPU if not then CPU) with has the packages mentioned above.
  - 

  - Browse the file 'Text Classification Competition.ipynb'
  - Once the 'Text Classification Competition.ipynb' is imported, please upload the test and train files by click on the arrow highlighted red below. If the arrow is not visible click on the folder icon on the left and then select the arrow:



- Please follow below link on how to import the file and run it on Google Colab GPU.
  - https://margaretmz.medium.com/running-jupyter-notebook-with-colab-f4a29a9c7156
- Once GPU is selected on Google Colab Runtime. click on Runtime on the menu and select "Run all" option as shown below



- Same has been explained in the video attached on the github.

4) Things I tried

I have tried various method as mentioned below:

i. I have done the preprocessing of the data like, removing stopwords, any character which is not alphanumeric, remove punctation. However with BERT only preprocessing I did was remove the following tokens (`'@user'`,`'..'`,`'<url>'`).
ii. I have tried various model, Navies Bayes (various variation like, MultinomialNB,ComplementNB,BernoulliNB), Logistic Regression, SVM, however with all of these I was not able to beat the baseline. I was revolving around .66 -.70. Out of all these BernoulliNB was able to provide better result.
iii. I have tried Neural networks as well but still not able to beat the baseline. I was still revolving around .66 -.70
iv. I have tried CNN with various filters and kernel but still not able to beat the baseline. I was still revolving around .66 -.70
v. I have tried LSTM with various combinations like, Bidirectional, different units (64,128,256) and also regularization but still it revolve around .68 -.71.
vi. In the end I have tried pretrained uncase base BERT model and was able to beat the baseline.
vii. I have tried various hyperparameter tuning for these models as well like:
    1) Smoothing parameter for Navies Bayes
    2) Tried TFIDF with various ngram_range
    3) Regularization L1 and L2 on deep neural network as well as other machine learning model.
    4) Used the word embedding 'glove.twitter.27B.100d.txt' as well as 'glove.twitter.27B.200d.txt' in neural network model.
    5) Tried adding dropouts, multilayer network in LSTM/BiDirectional LSTM.