NLP HW1

Ben Akhtar
906526538

October 7, 2023

1 ReadMe

The ReadMe can be found under the filename: README.md in the submission. Additionally, the following is the same replication information as in the ReadMe:

1.1 Instructions For Replication

1. Place the Jupyter Notebook file into google colab or an environment locally that has access to a GPU.
2. If you are running colab, you will need to edit the path to the folder in the first line. Additionally, in the 3rd code block you will need to change every single path to reflect where you first import the csv from, save the three ones created and then re-load them in as datasets. If not running in co-lab, these will need to be local paths, and the necessary commands may need to edited.
3. There is additionally a codeblock that contains "huggingface_hub" that will require you to have a hugging face API token to login. This will allow you to save the model and be able to run it. This will need to be your own.
4. All other commands should run normally. The last two codeblocks do not need to be run. This was just to create a PDF output.

2 Working with the Data

Working with the dataset required some extra work. Since it was necessary to split the dataset into 80%, 10%, and 10% for training, validation, and test respectively. Before this, the sentiments were re-labeled to labels and reviews to text. The text change was not necessary, but was easier to help follow the tutorial I used. The labels was to ensure that the model would be able to run. After this, the labels were replaced so that the model would be able to evaluate them properly. Positive was changed to 1 and negative to 0.

To generate the necessary splits This was accomplished by first loading the data into a pandas dataframe and then using scikit's built in train test split to split the dataset in the necessary ways. After this, each dataset was re-saved as a csv in the working directory. Then, these datasets were loaded into a dataset dictionary using the load_dataset module. This was to make it easier to work with for our trainer later on.

3 Building the Model

Most of the model building was followed using the tutorial found here: Pascual.

Once I have imported the necessary packages, I pre-process the data as described in the previous section. Next I separate the data dictionaries and load them into the tokenizer to create the necessary word embeddings for the Distilbert model. Distilbert actually has less work to do than regular BERT for the tokenizer section, which is partially why the speed is faster in general. After this, I create

tensors for our samples so the model can run in appropriate amount of time. I have then defined our model as Distilbert and used the function from Pascual to create some easier to read statistics once training and test are done.

I then create a place to save the model and then define our trainer function with the proper arguments to run the model. I ran on the training and validation datasets to start. Once the model is done training, I predict the results for every single split I have generated. The F1 scores fall in the acceptable range.

4 Analysis

The reason Distilbert was chosen, and this was a good tutorial to follow, was that Distilbert runs faster compared to regular BERT, but does mantain a high level of accuracy. This is because there is an overall reduction in the number of layers by about half. Additionally, the token-type embeddings and the pooler have been removed. Considering the use of Colab, this was an ideal solution. Additionally, since BERT is fine-tuned for sentiment analysis, it makes sense to use something that is derived from it to achieve a high level of accuracy and precision. It learns the context of the words and considering that movie reviews have a large amount of context, this is necessary for our model.

Other models that were tailored towards sentiment analysis would likely have achieved the necessary F1 score as well. They may not have been easily used, however, since they may not work as well with Colab and take far longer to train. For example, using a model such as BERT would have taken far longer to train and likely caused Colab to shut down during the middle of training the 40,000 reviews.

5 Evaluation

After training the model and the predict function was used on our model. This achieved the following results:
train: 0.9787392949611631
validation: 0.938126104890984
test: 0.9353153877059058

These results likely could have been better by increasing the number of epochs, however, this would have run into issues with Colab and the necessary scores were achieved.

References

Federico Pascual. Getting started with sentiment analysis using python. URL https://huggingface.co/blog/sentiment-analysis-python#4-analyzing-new-data-with-the-model.