Rohan Goel, Satej Shah, Paru Swaminathan
CS 410: Report

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

At a high level, the function of the code is to predict whether a piece of text (tweets and responses) is sarcastic or not sarcastic. Our program classifies the text (context) and associates a certain label (SARCASM or NOT_SARCASM) using an id. The input of our code is a json file named test.jsonl which contains responses with an associated id and context, and another train.jsonl file with responses, id, context, and a label. The output of our code is a comma separated file named answer.txt which contains the predictions on the test dataset. The file has exactly 1800 rows and each row has an id and the predicted label.

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 first step of our development process was to create a plan. Researching the different types of classification models, we were able to distinguish a couple of models that seemed promising to complete the task of classifying sarcasm of tweets. After testing some of them, we decided to proceed with the BERT language model and NN algorithm. Our first test was with FastText, however, the performance of this algorithm reached an upper limit, and therefore, we couldn't move forward with it. We also tried using K means with two centers, however we were unable to reach baseline with this. Therefore, we opted to go with using the BERT language model.

Using the pre-trained BERT model provided the feature vectors which we then used to train the Neural Network. We first used a Logistic Regression model, however, due to the simplicity of the model, it did not perform well enough. We then decided to use a Neural Network with 3 layers. Through tweaking parameters, we were able to finally get our average accuracy above the baseline.

In the code, we first read in the training and test data from the given folder. We then parsed the data. After this, we assigned binary values: 1 or 0 to the labels in the training data.

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.

To install and run this software, you can view the code through Github (https://github.com/rohang62/ClassificationCompetition). After cloning the repository onto your device locally, you can install Jupyter Notebook and open it through your browser. From here you can run the code blocks using the play button on the top of the screen. In the implementation of our code, we use many frameworks and libraries that may need to be installed locally such as PyTorch, numpy, pandas, transformers, and sklearn. After running all of the blocks of code, this will create a file "answer.txt" in which you can view the results of the program. Within this file, you will see a line for each situation and its resulting classification.

Brief description of contribution of each team member in case of a multi-person team.

Each member of the team learned and contributed to the project very well! Everyone was very enthusiastic, encouraging, and open to ideas. Working remotely, communication was very imperative to doing well. Rohan took lead on the development side, and Paru and Satej focused on the research and documentation. Although, all three members of the team helped each other out whenever needed. This team worked well together and was able to support each other throughout the course of the project.