Clarissa Hoyt and Roberto Rodas-Herndon

Final Presentation Essay

DS340: Introduction to Machine Learning and A.I.

Kevin Gold

Our project consisted of classifying text using the B.E.R.T language model created by Google. Our group had a common interest in text-classification so we decided that our main objective was to see how well a model could classify text that described a particular season (such as winter, spring etc.). We were particularly interested to see if Bert could accurately classify the data given a broad variety of different types of descriptions acquired from a survey passed around.

Our method of data acquisition consisted of mainly gathering survey responses from a breadth of different individuals in which each person had to provide a description of the 4 seasons of the year. We gathered around 200 or so records, turned the csv file of results into a Pandas dataframe data structure, cleaned it, and then began to work on fine-tuning our B.E.R.T. model. The first step to training the model is preprocessing. We downloaded the pre-trained BertTokenizer to tokenize each example of text in the dataframe, using a calculated variable for the max sentence length from our specific data as a parameter. Other parameters we chose to add was to pad the tokens to the max length and not truncate any data. Once all the data was tokenized, we had to split the data into training and validation sets. The sets were arranged according to the four labels, with a validation ratio of 20%, and a batch size of 3 because we do not have a large dataset. The next step was to download the actual Bert model. We used a pre-trained Bert model and AdamW as the optimizer. Before beginning training there needs to be some sort of function that can track the accuracy of testing the validation set. We chose a simple function that counts the true positives for each label and outputs the accuracy by dividing the sum of all true positives by the total number of

labels. The step of the actual training took a lot of testing with different numbers of epochs to find the right amount that reaches the highest validation accuracy without overfitting. The number of epochs we decided on is four. Now that the model is trained, it was time to test our results with some raw data.

The results were surprisingly good for the amount of data we had. Testing the model on new raw data gave accurate results for the majority of what we tested. We chose to officially evaluate our results by comparing them to a Bert model without fine-tuning. In order to do this we followed a tutorial that gave the results of a Bert model without fine-tuning by using a classifier on top of a pretrained model with the CLS vector as the feature vector. The resulting score was 63%. Our fine-tuned Bert model reached a 87% validation accuracy which means it was 87% accurate when running the model on the validation set after the training set. This shows that fine-tuning Bert gave a huge improvement for this dataset compared to not fine tuning.

Our results were suitable for the amount of data that we had available. Transformers usually work well with 1000's of records so we are fairly satisfied with what the model outputs. In fine-tuning the model, we realized that it worked best when using complete sentences rather than sentence fragments and the like.