Luke Monington

My Approach to Classifying Financial Statement Templates

My first step was to convert the data into a position that I could feed into different types of modes. To do this, I needed to find a way to convert the tags to numbers and ensure that all of the data was the same length. I accomplished this using a tokenizer to give a unique number to each of the tags then using pad\_sequences to trim/pad the sequences all to the same length. I saw that the average length of a sequence was 158.8 and the max length was 1,486. I trimmed/padded each of the sequences to be 300 numbers long, in order to avoid trimming too many of them and possibly losing valuable data but also avoiding over padding.

Next, I tried some ML models. I tried XGBoost and RandomForest. In order to get the best results, I used RandomizedSearchCV to randomly search among a list of parameters that I set out. Before getting into the results, it is important to note that the F1 score provides a harmonic mean of Precision and Recall and gives a better measure of incorrectly classified cases than Accuracy. After tuning, XGBoost achieved an accouracy around 98% (F1 score of 0.99) and Random Forest achieved an accuracy of around 95% (F1 score of 0.98). While this is pretty good, this just isn’t good enough to ensure that my submitted answers are all correct.

In order to improve my accuracy and F1 even more, and squeeze out the last little bit, I turned to a deep learning model. The layers in my model are: Embedding, Conv1D, GlobalAveragePooling, Dense(64), Dense(32), and Dense(1). I checked the accuracy of my model using StratifiedShuffleSplit to split my training data into 5 folds and to test the accuracy on the cv data for each fold. From this, I was able to consistently achieve a val\_accuracy of at least 99.5%, but usually 100% and F1 score of 1.0. This looked to be promising, so I trained my model on the full dataset and converted the final predictions into a csv file.