Kevin J. Bird
11/28/17
HackerEarth: Predict the Happiness

Approach

I imported the fastai library which is an open source library built on top of pytorch. The first step was to split each review into separate files. This was helpful to get each of the words. Once this was done, I built a model that used Adam as the optimizer, size 200 embedding vector, 500 hidden activations, and 3 layers. These are all hyper-parameters that would be better to optimize if there were more time.

At this point all I was doing was getting weights for my embedding matrix. This was used later on for the final model.
The final model was created using 3 recursive neural networks. The tough part is mostly taken care of by the fastai library. This is the command that creates the model:

m3 = md2.get_model(opt_fn, 1500, bptt, emb_sz=em_sz, n_hid=nh, n_layers=nl,dropout=dropout, dropouti=dropouti, wdrop=wdrop, dropoute=dropoute, dropouth=dropouth)

I actually played around with the values, but up to this point, the default values have been the most successful that I have found. The only value I actually had to worry about was emb_sz, which had to match the embedding size from the previous model. Once each of these models was built, I played around with fitting just the top layer and also fitting all of the layers. Each of these models was selected and loaded with the embeddings from a different point in the training cycle from the earlier word prediction model. Once these three models were all trained and the models were saved, the file was created. Basically each of the predictions was build independent of the other two and then the scores were all added together. This gave me a 0.90906 score on the public leaderboard.

Lessons Learned

1. The embeddings created for the word prediction model were extremely important and I had very large fluctuations on my final result when I would try to save time and undertrain these. My assumption on why this was so important is that since this was the foundation of the final prediction, if there was a problem with the initial embeddings then things weren't able to be built on top of that foundation that transferred the knowledge to happy/not_happy.

2. Save everything often and know exactly what you are saving. There was a long time that I was loading the embedding weights into my model and not realizing that I still needed to train my model so my scores were all over the place depending on whether I was building a new model from scratch or trying to use weights already gathered previously.

3. Make one change at a time once a good score has been achieved. There were way too many times that I would try to reinvent the wheel and get myself in a worse case without really understanding what had changed to give me the worse score. Was it the new number of cycles? Dropout rates? Embedding size? I needed to change each parameter slowly to be able to say more definitively one thing works or doesn't work.

4. Roll with the mistakes. One of my earlier mistakes that I made ended up helping me stumble into the fact that a very well trained word prediction model will give you a good final score. I

misunderstood what a certain function did and basically I had told my model to train a total of ~1.2M epochs at 2 minutes each (I thought I told it to do 40 epochs). When I came back to check on it the next day, it was on epoch 442 and I thought there was a bug in the fastai library. After digging into it further, I found my issue, but more importantly, I had the best embeddings I had used. This helped me get my initial best score and also gave me a much better understanding of how important the word prediction model is for the sentiment prediction.

5. It is impossible to try every tuning possibility for hyper-parameters. Instead of trying to tweak everything slightly, go for extremes and see if any patterns are identified. Think your dropout is too low? Crank it way up and see if it does what you think it should. If it does you can drop it back down until you get to a more reasonable number, but if it doesn't, maybe you actually need to look at some other numbers.

6. Fastai is a really powerful library that provide a lot of flexibility and gives a lot of good assumptions. It really does feel like the type of library that is going to change the way people in industry look at data science. Instead of a select few people having the knowledge to build predictions, it will allow the subject matter experts to build the models themselves and use their own intuition to build the best model for their problem.