Capstone 2 – Second Report
July 27

For this milestone update, I experimented with different datasets and model architectures to attempt to improve the predictions and scores for the machine translation component of the project. As the model stands in its most recent form, it generally stops improving on the validation data after about 10-15 epochs of training, and the accuracy score on this data generally does get very far above 60%. The training data does a bit better, with continued improvement throughout all epochs, and reaching accuracy of around 70-75%. These are clear signs of overfitting, since the model essentially stops learning and adjusting itself to improve on the validation set. Testing different datasets and architectures has helped me to learn more about the reasons it is overfitting, as well as rule out unhelpful solutions to the problem. For the datasets, I tested both longer sentences, as well as parsed the original Chinese data differently than I had previously. For model architectures, I tested varying neuron and dropout numbers, as well as additional layers.

1. Testing Different Tokenization Methods

One aspect that stood out during data exploration was the number of singly-occurring words in both the English and Chinese sentences. We can expect problems around these words because if a word that only appears once occurs in the testing data, we can be sure that it was not learned during training and the translation, if it translates at all, will be inaccurate. I noticed that there were more than double the number of singly-occurring words in the Chinese sentences (51% of all vocabulary) than there in the English sentences (40% of all vocabulary). To attempt to decrease some of the noise in the data and reduce vocabulary size in general, at least on the Chinese side, I postulated that I might be able to tokenize the Chinese sentences into unigrams.

Chinese words can be one or more characters in length, and the same characters can occur in different combinations. For example, putting aside whether the usage is common or not, the words 合 (whole) and 適 (proper) can be also be combined into the words 合適 (suitable) and 適合 (to be suitable), comprising four individual tokens. However, if we split at unigram level rather than at word level, we only have two tokens. The Jieba package, which I have been using, splits by words of one or more characters. But it is worth checking to see if splitting by unigrams instead might decrease the vocabulary size, thereby enabling us to use more data, as well as improve the translation results, since there would be fewer singly-occurring characters in the testing set.

After splitting by unigrams, even with the data increased by 3,000 sentences, the total Chinese vocabulary size (number of unique words in the dataset) decreased from 8,600 to 2,590, although average sentence size increased from 5.1 words to 8.4 words. Singly-occurring words decreased from 4,422 to 434, from 51% of the vocabulary down to 17% of the vocabulary. During model training, training scores did not show much improvement over scores from word-level tokenization, although validation scores improved. In general the BLEU scores for both the training and testing sets were not better than with the prior tokenization strategy.

Subjectively speaking, the randomized translation printouts for the training sets seemed similar to the previous attempts, but the translations of the testing data seemed slightly better.

2. Testing Longer Sentences

The dataset of longer sentences comes from the University of Macau corpus. The sentences are taken from the spoken language set, since the aim of this project is to eventually add an audio component to translate speech. Overall, this attempt was not successful. The longer sentences were computationally expensive, especially during sequence encoding. In fact, the RAM usage kept causing Google Colaboratory to crash, and in order to have a successful run through, I had to reduce the number of sentences down to 2000. With such little data, even when applying both word level and unigram tokenization, the results were unsurprisingly bad. All sentences translated as "the the the the the the the the the the the the," exactly eleven "thes." Although the English sentence length average in this case was 19.6 words, the minimum sentence length was 10, and sentence with 11 words also occurred quite frequently. The word "the" is the most commonly occurring word in the English data.

3. Adjusting Model Architecture

The adjustments made to the model include testing varying neuron and dropout numbers, as well as additional layers. From further reading I learned that dropout layers actually dropout a specified portion of the input or output during training, and the argument recurrent_dropout in the LSTM layer induces dropout in the network neurons. Both types work to prevent overfitting, since the former works to prevent memorization by increasing randomness in the data seen by the model, and to prevent neuron weights from becoming fixed too quickly. In addition, I also tried changing the original LSTM and attention layers to two Bidirectional LSTM layers. The results in terms of accuracy, BLEU scores, and translation quality did not improve, but it showed promise in terms of reducing overfitting. In the previous iterations, the model would stop improving on the validation data at a certain point, but with these adjustments it was able to continue to train longer with continuous – if more incremental – improvement. It was able to continue to improve even after 60 epochs, although by that time the training graph did show signs of improvement slowing.

4. Next Steps Summary

At this point the biggest issue seems to be finding a way to increase the amount of training data. One of the tradeoffs with using Google Colaboratory instead Jupyter Notebooks is that while I do get access to a GPU runtime that significantly reduces training time, there are limits to the RAM. In other words, there are limits placed on the size of the data because if it is too large when performing an operation, then the session will crash. The was especially obvious with the long sentences test. Google offers options to pay for more RAM, so I will explore the costs of that to see whether it is worth spending some money for a better training session.