Attention Based NMT with LSTMs

Joseph Suarez

Abstract

As of present, this document serves as a catalog of experiments performed in the process of implementing attention, along with a mix of quantitative and qualitative results thereof. Despite having a complete implementation of attention that, by all experimental results, appears correct, a few technical issues prevent us from conducting large scale training.

Introduction

Our attention implementation currently implements separate train and test routines and includes the ability to efficiently sample translations and evaluate BLEU score. Our architecture is the same as described by Luong et al, with a global attention model. In the process of attempting to reproduce the results presented there, we evaluated a large space of hyperparameters. This served, and continues to serve, as both a mechanism of diagnosing potential implementation errors (none were found) and better understanding the particular hyperparameter configurations that cause train or test time problems. We examined:

- Gradient descent vs. Adam
- Adaptive learning rate, learning rates between 0.00001 and 1.0
- Number of hidden units between 32 and 256 (for speed of dev cycle)
- Number of LSTM layers between 1 and 4
- Batch size between 1 and 16 (larger batch sizes did not cause memory issues when using experimental gradient aggregation)
- Number of training examples: overfitting 1-2048 examples, training on 130k English-Vietnamese.
- Activating/deactivating attention

Our first course of action was to strip the model down to the bare bones until we found a guaranteed-working model. This meant removing attention, training on a single data sample, tweaking the learning rate, and using a single LSTM layer.

We then began increasing model complexity in order to evaluate the correctness of various components. First, we tested for errors in the various calls to reshape that could cause the model to be correct on one sample but incorrect on multiple (e.g. evaluating softmax along the wrong tensor axis). We successfully overfit 4 and 16 examples (and the predicted translations were correct), thus this could not be the case.

Next, we attempted to increase the number of LSTM layers. This made training highly...finicky. Learning rate had to be carefully tuned, using both Adam and minibatch SGD, to avoid diverging (or static) training error. After a large number of tests with mixed success, we backed off on this frontier and moved on to testing other components.

Next, we tested whether attention prevented successful overfitting. It did not, and train error decreased as expected on tests with more training examples. This suggests the correctness of our attention implementation.

Next, we notice that in some of our tests, the error plateau corresponded to predictions of all “STOP”. This makes sense intuitively, as the sentences are padded with stop tokens up to length

30. Thus, we conduct a somewhat larger experiment to determine whether this data representation is the source of poor convergence. We re-preprocess the dataset using only sentences of length at least 30. Thus each sentence has a single start and stop token. We then repeat the tests above and evaluate translations. The error then plateaus, predicting all commas, or in some cases the same repeating common phrase. In smaller overfit tests, this first phase was succeeded by a slightly better separate phase. Namely, some of the translations were correct, and others were duplicates of the correct translations on unrelated sentences. After further training, the overfit test eventually learned to correctly predict the 16 sentences with BLEU score 100. As this dataset only contained a few thousand sentences and offered no advantage over the original dataset of 130000 sentences, we return to the original format.

We then run a larger overfit test with 2048 sentences with 1 LSTM layer, 256 hidden units. The cross entropy decreases to 3.0 after 8 epochs, but the translations resemble the repeating phrase plateau of the smaller overfit tests. However, we are unable to get past this plateau.

At this point, we have mostly exhausted the set of hyperparameter configurations; at the least, we should have found one that performed reasonably. To the best of our judgment, this leaves only two possibilities. First, we could have an error in our implementation that we simply have been unable to find. However, none of our tests point to a particular erroneous component of the algorithm (the addition of multiple layers to the LSTM is a two line tensor flow drop in that is a highly unlikely place for error). Second, our tests could simply not be running for long enough. Thus, we overnight a final test using a set of parameters similar to those in the original paper: 256 hidden units, 4 LSTM layers, 128D embeddings, minibatch size 30, Adam with eta=0.001. After 15% of an epoch, we reach train CE of 3.0. After 1.25 epochs, we reach train CE of 1.8. However, BLEU score remains 1.81. Will continue training for some time.