Zero-Shot Translation

Jeremy Irvin

1 Summary

In Google's recent paper 'Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation', a single model is able to produce good multilingual translations, often when the tested language pair was never explicitly seen during training. This model is a simple extension of previous NMT models which use multilayer LSTM encoder (with a bidirectional first layer) and decoder with attention to translate variable-length sentences of different language pairs. This model proposes a simple change to the data pipeline, namely the addition of a single language token in every source sentence which indicates which target sentence to translate into (the use of this token in this way is learned by the model automatically) as well as a shared wordpiece vocabulary. This wordpiece vocabulary is created with a greedy approach where the tokens begin as characters and are merged, pairs of tokens at a time, based on their frequency in the training set (the most frequent pair gets merged in every step). One specifies the number of wordpieces (the number of merges), which is typically one of 8K, 16K, or 32K, with 32K demonstrating the best results in this paper. The model is then training jointly on all the language pairs with this shared wordpiece vocabulary and special language tokens inserted for several epochs until convergence.

2 Experiments

I have run many experiments over the past few weeks, most of which have resulted in low quality translations. I have implemented a bidirectional first layer in the encoder, a projection layer between the encoder and decoder to pass the forward and backward hidden states into the decoder, dynamic rnn in the entire encoder, and the whole data pipeline with french-english preprocessing and wordpiece vocabulary generation and splitting (this part was modified slightly from an existing implementation). My initial experiments were run using varying number of wordpieces on 100K-200K sentence pairs of each of the french-¿english, english-¿german sentence pairs (unidirectional only), with different architectures including with and without the bidirectional first layer, with and without the hidden state projection between the encoder and decoder, and different number of layers and LSTM cell sizes. My models were able to achieve low cross entropy loss but zero shot translations were very poor. In order to try to improve this, I recently began training models on the full dataset (2M sentence pairs of french-¿english and english-¿german for much fewer epochs). The translation qualities on the individual languages seemed to improve much faster, but never reached good quality. I believe this is due to a lack of model complexity because of the small RAM on the GPUs, which limited my models to a maximum of 2 layers with 512 cell sizes with a 16K wordpiece vocab. Google's paper runs a model on a similarly sized dataset using 8 layers and 1024 cell sizes on a 32K wordpiece vocab. The CE loss drops quickly but plateaus at around 4.6, and the translations are poor but not complete gibberish. I have tried different optimization techniques which have been unsuccessful in optimizing this further.

A few things I noticed in my experiments is that a bidirectional first layer is important in allowing the model to use the language token in the source sentence to identify which language to translate into. Models without the bidirectional first layer would almost always translate into a single language. Moreover, passing the hidden state significantly improved translation (without the projection to speed up training times). When zeroing this state out, the model outputs the same start token at every sentence. When passing it, the start tokens differ (and are mostly correct).