PKUWriter

The OpenNMT is a fine-tuned seq2seq model. The baseline based on OpenNMT is good and challenging.

Apart form using OpenNMT seq2seq model, we realize a seq2seq model with Tensorflow(1.2.1) seq2seq API, which includes bidirectional encoder, Luong attention, and dropout layers. To be more specific, we have one encoder layer and one decoder layer. The dropout rate is 0.3. All the sizes of hidden layers are 800. The max size of a batch is 64. And the learning rate starts from 1.0 and decayed by 0.7 after every epoch, using Gradient Decent Optimizer.

To make our model better, we apply reinforcement learning in addition to the basic seq2seq. As we find that one of the most important things in the generation of this task is that you should make most subjects get their descriptions in the generated lex, which means that the subjects occuring in the triple set should also occur in the generated texts. And we develope a reinforcement learning algorithm to approximate this goal, which makes about two more BLEU points on performance.

We find that it is rather hard for a single generation model to perform well on every triple set. Nonetheless, if we train multiple generation models and somehow have the ability to select the best one out of the results generated by those generation models for every given triple set, we will witness a notable increase in the evaluation scores.

In order to do that, we used a technique called Learning-to-Rank, which has been widely used to rank the quality of a relevant documentation given a query. Here we use it to evaluate the quality of the result generated by the generation models given a triple set.

The details can be described as follows. First of all, we divide our training data into two separate datasets, dataset A and dataset B. N generation models are trained on dataset A and for triple in dataset B, we generate predictions from the n generation models. For every pair of <triple set, prediction>, over 30 features are extracted to characterize the quality of

the prediction, including word-level features, sentence-level features and readability-related features. The quality of this triple-prediction pair is defined as the BLEU score between the prediction and the expected lexicalization, which is used as the ranking score of this triple-prediction pair. For every triple-prediction pair, we have now acquired a list of features and its ranking score. Then we use these features and ranking scores to train a Learning-to-Rank ranker. The ranker, after trained, will be able to predict the ranking score from the features of a given triple-prediction pair. Here we use LAMBDAMART, a robust and powerful model, as our ranker. Now for any triple set in the test dataset, we can acquire n triple-prediction pairs from our generation models. Then features are extracted and the ranker will decide which prediction is the best for every triple set. Considering that the performance of the ranker is not completely stable, we employed multiple rankers and choose the prediction that most of the rankers consider the best. The performance of this ranking system is stunning. In our experiment, we were able to achieve a BLEU score of 58.02 on the given dev dataset out of 7 generation models(including official baseline model), the best of which can only achieve a BLEU score of 53.40(this score shrank a little due to the fact that part of the train dataset was used to train the ranking model).

When generating predictions for the given test dataset, we use the given train dataset as training data for generation models and the dev dataset to train our Learning-to-Rank model. Besides, for the inputs that our model cannot handle well, we have manually checked the outputs and added some handcraft rules, which are primarily used when making predictions.

Contacts:

Xiaojun Wan (wanxiaojun@pku.edu.cn)

Hongyu Zang (zanghy@pku.edu.cn)

Liunian Li (liliunian@pku.edu.cn)