CSE-443: Natural Language Processing

Machine Translation Quality Estimation

Introduction

Machine Translation quality estimation (QE) refers to the task of measuring the quality of machine translation system outputs without reference to the gold translations. There are mainly three variants of the problem: word level, sentence level, and document level. In this work, we focus on word-level machine translation quality estimation. The QE is gaining popularity as they reduce post-editing time and also results in saving in labor costs.

The word-level quality estimation task can be framed as a binary classification of each translated token assigning one of the two labels: OK and BAD. Earlier work focused on using handcrafted features and then using some simple regression or classification methods. Recent results use recurrent neural networks for the same as they help in capturing global content. However, these models encode the context words by concatenating only the left and right words giving them limited ability to control the interaction between local context and target word.

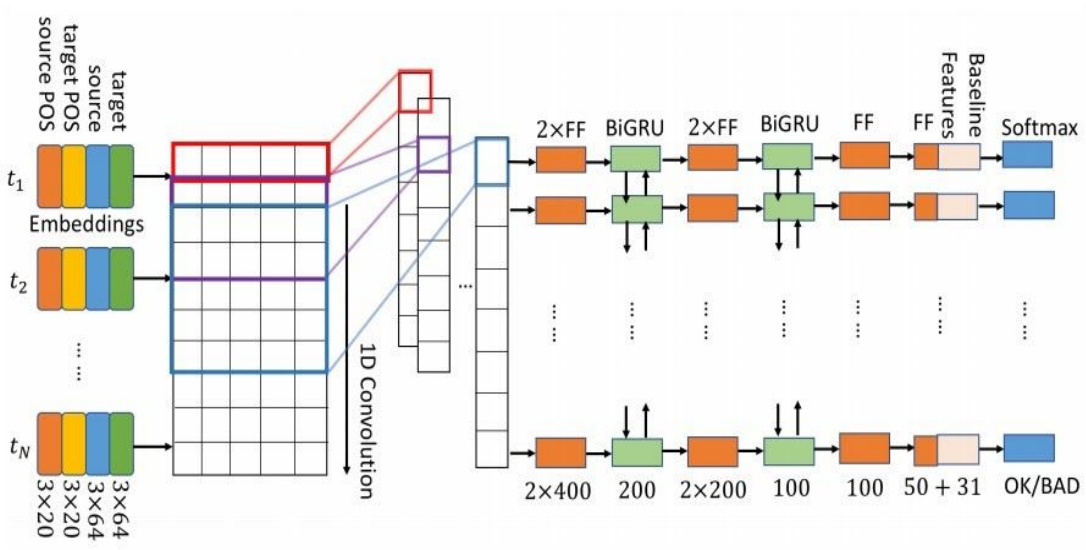
In this project, we combine the approach which utilizes both handcrafted features and recurrent neural networks to make decisions. Our model consists of a three-part neural network:

1. An embedding layer to represent words and their part-of-speech tags in both source and target language.
2. A one-dimensional convolution layer to integrate local context information for each target word.
3. A stack of feed-forward and recurrent neural networks further encode the global context in the sentence before making the predictions.

Data

The training data consists of source and machine-translated sentences from English to german. In addition to the sentences, word-to-word alignments were also provided in the data. The training data consists of 13,442 sentences, whereas the test split of the data consists of 1023 sentence pairs.

Model



The model consists of an embedding layer where the source and target words are embedded. The embedding of the neighboring words is also concatenated with the current word. Now 1D convolution is used along the time axis. We use various sizes of convolutions: 1,2,3,5, and the output of all is concatenated. Appropriate padding is applied so that the number of output embeddings is the same as input embeddings.

Now, these output embeddings are passed through a stack of fully connected layers and bidirectional GRU, and finally, a softmax layer is used to get the probability of each class. We use binary cross-entropy loss as the loss function and use adam optimizer with a learning rate of 0.001, which decays linearly with the number of epochs.

In our other experiment, we have used LSTM instead of GRUs and removed all the models' baseline features. Through this, we try to train the model in an end-to-end fashion. The results were similar to the previous experiments, using GRUs and baseline features.

Results

We train our model for the english-german QE task where the source sentence is in English and the translated sentence in German and report the results:

Metric	Score
Accuracy	82.68
F1 (macro)	55.38
F1 (micro)	82.68
F1 (weighted)	80.49

Table 1: Performance of the model on the test set trained using GRUs and Baseline features

Metric	Score
Accuracy	81.27
F1 (macro)	56.12
F1 (micro)	81.27
F1 (weighted)	80.02

Table 1: Performance of the model on the test set trained using LSTMs and without Baseline features

We observe that since the classes are imbalanced (a large number of OK labels compared to BAD), our model overfits and is easily biased towards the majority class (OK). The GRU performs comparable (slightly better) than LSTM. This is because LSTM has more parameters than GRU; it is more likely to overfit than GRU. Having a more extensive dataset can help us reducing the overfitting in the model.

Future Work

For future work we plan to do the following:

1. Currently we are using an embedding layer for our model. But due to the small size of the dataset, the embeddings learned may not be a good representative for the word. To tackle the embedding we plan to use pre-trained embeddings from fasttext or BERT and also experiment with BPE(Byte Pair Encoding).
2. The model overfits very quickly due to large data imbalance. We plan to use regularization techniques and data upsampling to tackle this issue.