DL2 Report

Weiguo Ye

May 6, 2019

1 Model

As a baseline model, I used an embedding layer, a one-layer bidirectional RNN to better capture the context information for each time step, and a dense layer into which the output of RNN is directly fed. A residual connection from RNN input to the output is used to help training and boost performance. And I applied an extra CRF(Conditional Random Field) layer on top of the outputed logits, which is similar to HMM, with an extra transition matrix to train that may impose tag transition rules. After all, I used Viterbi-like decoding process to find out the most propable tag sequence. I found using different configurations for different languages is helpful, so I used num_terms to deduce the language type.

1.1 Configuration for Japanese

GRU with 50 hidden units is used for Japanese because it requires less parameters. The embedding size is also 50 so that it can directly be fed into GRU.

1.2 Configuration for Italian and Hidden Language

It looks like Italian and the hidden language are more complicated and requires more complicated model. I used LSTM with 100 hidden units and a embedding size of 70. To fit into the size of LSTM, the embeddings will first be passed into a dense layer to produce a tensor whose last dimension is of size 100. The embeddings are also concatenated to the output of LSTM before the last dense layer, which can help model make decision simply based on the embeddings.

2 Finetuning

I used Nadam optimizer to train the model, which is a nestrov version of Adam. Considering] the training time required is extremely limited, I have to use a decaying learning rate with relatively large initial value and raise keep_prob for dropout. I utilized consine decay strategy, which unlike exponential decay, has an increasing decaying rate thus gives larger learning rate before end. I

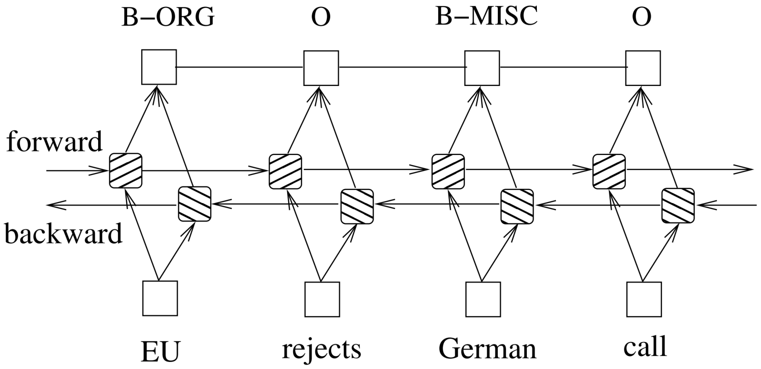

Figure 1: Bidirectional RNN-CRF Model (Zhiheng H et.al., 2015)

only tunes the leanring rate and decaying step (number of steps to reach π for consine). The setup is (0.01, 1400) for Japanese, (0.017, 1350) for Italian, (0.015, 1400) for the hidden language. I find directly giving up dropout gives higher accuracy for Japanese and keeping large keep_prob (0.95) for Italian. I didn't try a lot random seed, but different random seeds can have significantly different results. The model with those configurations can have +0.2 0.3 results in my local test with the same number of iterations. The grading script seems to work differently in terms of random number generation even with the same random seed.