Assignment 2 : BiLSTMs

Kartik Sathyanarayanan (EID: ks46373)

March 15, 2018

1 Introduction

Part of speech (POS) tagging is a fundamental task in natural language processing which helps in understanding if a particular word is used as an adjective or adverb or does it belong to present tense or past tense. The problem can be easily solved through Simple Recurrent Network which can incorporate contextual information from long period. But, in traditional simple recurrent neural network (SRN), during back-propagation, gradients are multiplied with the weight matrix corresponding to the neurons of the hidden layers. If the weights in the matrix are small, then the gradient for the neurons becomes too small and it will be very hard to train the SRN (vanishing gradient). Similarly, if the weights in the matrix are too high, then the gradient for the neurons in the hidden layer will be too high and causes the learning to diverge (exploding gradients).

LSTMs overcome the issue of vanishing/exploding gradients by maintaining a cell state. Each cell state consists of three gates - input, output and forget. Input gate is used to determine how much of the new information in the current time step we're going to store. Output gate determines how much information in the current time step we'll use in the next time step. Forget gate decides how much information we wish to remember from previous time steps. In BiLSTM, there are two LSTMs which process the input sequence in forward and backward direction and hidden layers at each time step are concatenated to form the output for the cell. It has proven to perform very well on part-of-speech (POS) tagging task. In this assignment, we use orthographic features along with word embedding for POS tagging.

We check the accuracy of the model by including the orthographic features as one-hot vector in two ways. First, we add them in the existing input layer as additional one-hot features. And in the next way, we include them as features in the final POS classification layer of the network. We also calculate the out-of-vocabulary (OOV) accuracy for the BiLSTM.

2 Methodology

Given a sentence $w_1, w_2, .., w_n$ with tags $y_1, y_2, y_3, ., y_n$, BiLSTM is used to predict the tag probability distribution of each word. There are two types of features used for predicting the POS tagging. First, w_1 is the word which is embedded into lower dimension. Second, Orthographic feature (w_i) are extracted for each word which contains a common English suffix (e.g. -ing, -s, and -ly), contains a hyphen, and starts with a number or capital letter. For complete list of suffixes, refer table 1. These orthographic features are converted into one hot vector by using the inbuilt function create one_hot() of tensor flow. These orthographic one hot features which are used to predict the POS tags are passed in two ways in the model which will be discussed in the next subsection.

BiLSTM incorporates the information from the past and future while predicting the pos-tag for the current word. The output layer of the BiLSTM is passed through linear transformations to get unnormalized scores. Then, the output of the linear transformation is passed through the softmax layer whose dimension is the number of the tags. It produces the tag probabilities distribution of the word, w_1.

suffix	'acy', 'al', 'nce', 'dom', 'nce', 'er', 'or', 'ism', 'ist', 'ty','ment', 'ness', 'ship', 'ion', 'ate', 'en', 'fy', 'ize', 'ise', 'ble', 'al','al', 'esque', 'ful', 'ic', 'ical', 'ous', 'ish', 'ive', 'less', 'y', 'ship','ary', 'hood', 'age', 'logy', 'ing', 's', 'es'

Table 1: Suffix list which is used as an orthographic feature

2.1 Orthographic Features as input

One-hot orthographic feature vector is concatenated with the input word embedding and passed as input to the BiLSTM which can be clearly seen in figure 1.

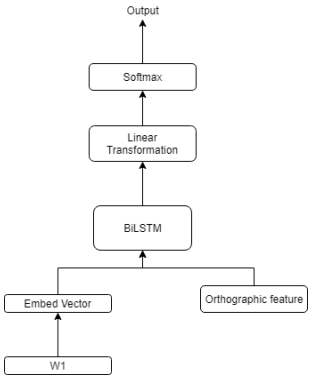

Figure 1: Model 1 - BiLSTM model in which orthographic features are concatenated with the input

2.2 Orthographic features as input to linear transformation

One-hot orthographic feature is concatenated with the output of the BiLSTM and passed as input to Linear Transformation as shown in figure 2.

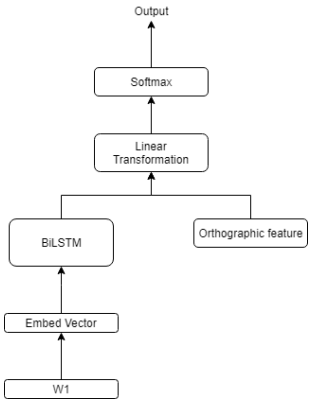

Figure 2: Model 2 - BiLSTM model in which orthographic features are concatenated with the output of BiLSTM

2.3 Out of Vocabulary Accuracy

We have also captured the out of vocabulary (OOV) accuracy so that we can check the effect of orthographic features on predicting the POS tag for the out of vocabulary (OOV) words. To calculate the out of vocabulary accuracy, we created a mask which check which words have id equal

Model	Accuracy (train)	Accuracy (val)	Accuracy (test)	OOV Accuracy (val)	OOV Accuracy (test)	Training runtime (hh:mm:ss)
Model-0	97.72	95.5	95.6	54.74	53.8	00:06:36
Model-1	98.02	96.4	96.3	77.12	75.7	00:08:42
Model-2	97.80	95.8	95.8	60.44	57.7	00:07:32

Table 2: Experiment results

to the length of the training set vocabulary size. The words which have this assignment are our OOV words.

3 Experiments

We train and test our model on Wall Street Journal data from PennTreeBank usinga split of 80% for train, 10% for val and 10% for test. We calculate the accuracy, validation accuracy, out of vocabulary accuracy. The vocabulary we used in the experiments are the words that are present in the WSJ Penn TreeBank training set and one UNK symbol for replacing out of vocabulary words

3.1 Accuracy analysis

Table 2 compares the performance of different models with the baseline. It is of no doubt that without using orthographic features, OOV accuracy and overall testing accuracy are very less.

Orthographic features boost up the **performance** of the model accuracy from 95.5% to 96.4% if we use the model 1. The **reason** is that these orthographic features are helping the model to predict the correct label that have same orthographic features, for e.g. words ending with -ing are mostly verb, capitalized words are mostly Proper Nouns. Also, the **OOV accuracy** increases from 53.8% to 75.7%. It shows that the baseline model is facing difficulty in correctly tagging the words. Using these orthographic features, it can decide the correct POS tag to some extent.

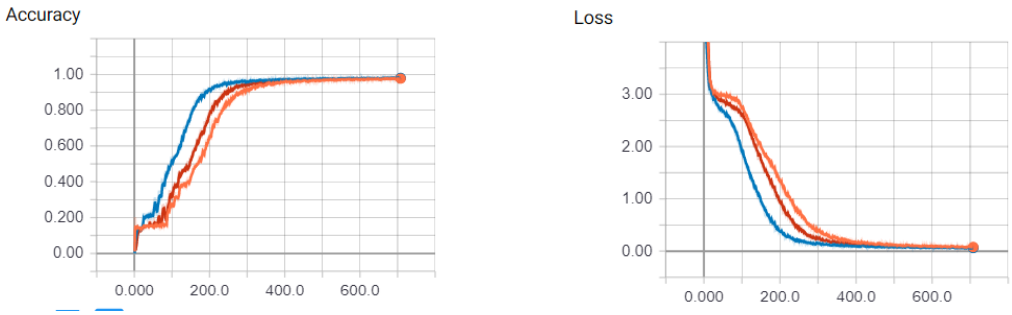

Figure 3: Training accuracy and loss from Models 0, 1 and 2. Orange - Model 0 ; Blue - Model 1 ; Red - Model 2

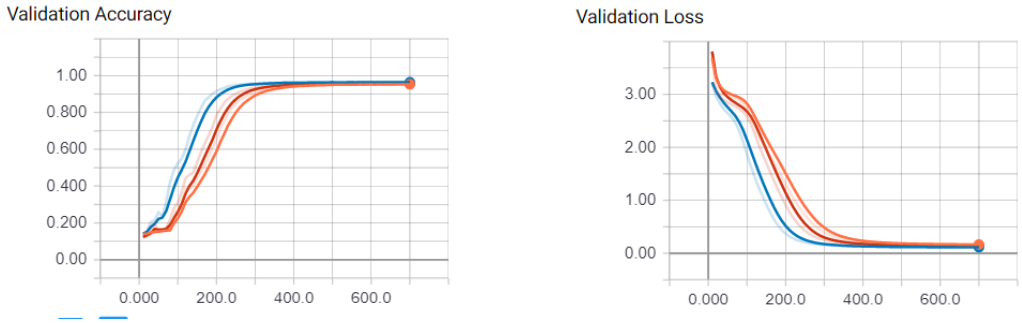

Figure 4: Validation accuracy and loss from Models 0, 1 and 2. Orange - Model 0 ; Blue - Model 1 ; Red - Model 2

From figures 3 and 4, we can see that training and validation accuracy for model 1 boosts up pretty quickly as compared to model 2 and baseline (Model-0). Model 1 achieves saturation after 250 iterations, however, other models achieve it after 450 iterations. Model 2 and baseline model have almost the same convergence path which shows that model 2 did not impact training and validation accuracy or loss much. Similarly, from the training loss and validation loss, we can see that model 1 loss decreases faster compared to model 1 and baseline loss (almost same loss for both model). This shows that passing orthographic features to the BiLSTM input helps in achieving training and saturation of the model faster compared to the other models.

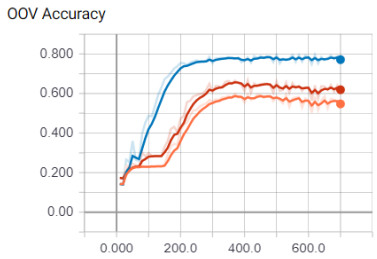

Figure 5: OOV Accuracy of all models.
Orange - Model 0 ; Blue - Model 1 ; Red - Model 2

From the figure 5, we can see that OOV accuracy for the model 1 boosts up pretty fast as compared to the model 2 and baseline. Model 1 achieves saturation after 300 iterations, model 2 achieves it after 400 iterations whereas baseline achieves it after 500 iterations. Similarly, from the table 2, we can see that model 1 boosts up the overall accuracy and OOV accuracy to a larger extent as compared to model 2. The reason is that in model 1, the contextual information from orthographic features are encoded in the BiLSTM for a long period of time which is very useful in predicting the POS tag, for e.g. adverb modifies the verb, adjective and other adverbs. However, this "long distance" dependency is not incorporated in model 2 as orthographic features are directly passed to the transformation layer which is using the orthographic features for the current word tagging (dependency no longer exists).

3.2 Training time analysis

All experiments were run on Nvidia GPUs in the eldar cluster of the department. Let's compare the training time taken by all the models. Model 1 takes the longest time to train as shown in table 2. This is because network size is larger and it needs more time to train the model. Baseline has the least number of parameters in comparison to model 1 and model 2. So, the Baseline (model-0) takes the least time to train.

Model 1 takes longer time than Model 2 because in model 1, increase in input tensor size increases the number of weights required for the LSTM gates in comparison to the addition of orthographic vector in the linear transformation layer in model 2.

4 Conclusion

In this work, orthographic features with word embeddings are used to train the BiLSTM model. We check the accuracy of the model by including the orthographic feature as one-hot vector in two ways. In first way, we add them in the existing input layer as additional one-hot features. And in the next way, we include them as features in the final POS classification layer of the network ,i.e., the current hidden layer and the orthographic features are used to predict the current POS tag. The results of our experiments shows that though we didn't get much improvement in the accuracy for the model 2 which is expected but we get very high overall accuracy and test accuracy for the model 1 because of the contextual orthographic features learned by the BiLSTM model.