Part-of-Speech Tagging with Bi-LSTMs

Nayan Singhal (ns28844)
University of Texas at Austin
Austin, TX 78712
nayans@utexas.edu

Abstract

Bidirectional Long Short-Term Memory has been shown to be very effective for pos-tagging. Though word embedding has been shown as powerful for statistical pos-tagging of the sentence. In this task, we use orthographic features with word embedding for part-of-speech tagging. When tested on the Wall Street Journal, accuracy: 96.3% and out of vocabulary accuracy: 71.4% is achieved.

1. Introduction

Part of speech is a fundamental task in natural language processing and it helps in understanding the particular word whether it is used as an adjective or adverb or does it belongs to present form, or past form. The problem can be easily solved through Simple Recurrent Network which can incorporate contextual information from long period. But, in tradition simple recurrent neural network, during back propagation, gradients are multiplied with the weight matrix corresponds to the neurons of the hidden layer. This means that weight matrix have a great impact on the learning process.

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 which is known as vanishing gradient. Similarly, if we have weights in the matrix too high, then the gradient for the neurons will be too high which causes the learning to diverge, known as exploding gradients. Because of vanishing and exploding gradients, its very hard to learn the model.

These issues are the main motivation for LSTM in which 3 modules have been introduced for each cell. Each cell consists of three parts: input gate, forget gate and output gate. Forget gate decide what information we are going to throw away. Input gate decided what new information we are going to store in the cell state. And, output gate decides which information we are going to output to the connecting cell. And, in BiLSTM, separate LSTMs process sequence forward and backward and, hidden layers at each time step are concatenated to form the cell output. It has already proven a powerful method for sequence labeling.

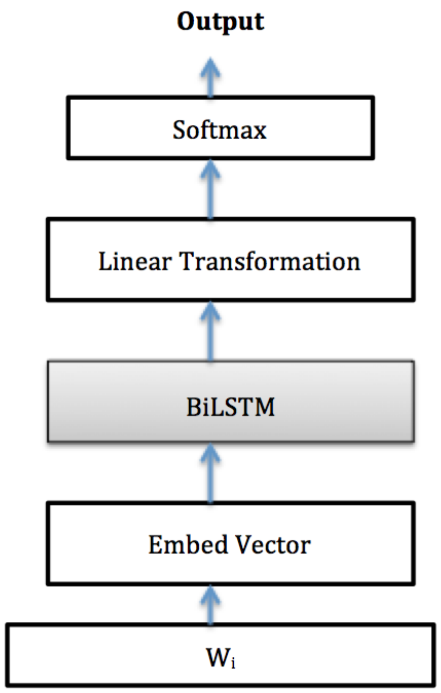


Figure 1: BiLSTM model for pos-tagging

The main contribution of this work include: First, 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 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). Second, it shows how to calculate the out-of-vocabulary 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.The complete model is shown in fig:1. There are two types of features used for predicting the pos-tagging.First, w1 is the word which is embedded into lower dimension. Second, Orthographic feature(wi) are extracted for each word which contains a common English suffix (e.g. -ing, -s, and -ly), prefix (e.g. anti-, auto-, semi-) contains a hyphen, and starts with a number. For complete list of orthographic features 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 are used for predicting the pos-tag but they are passed in two ways in the model which will be discussed in the next subsection.

suffix	acy, al, nce, dom, nce, er, or, ism, ist, ty, ment, ness, ship, ion, ate, en, fy, ize, ise, ble, al, esque, ful, ic, ical, ous, ish, vie, less, y, ship, ary, hood, age, logy, ing, s, es
prefix	a, an, ante, anti, auto, be, circum, col, com, contra, contro, con, co, deca, de, dia, dis, di, en, extra, ex, fore, hecto, hemi, hetero, hexa, hepta, homo, homeo, hyper, il, im, inter, ir, intro, intra, in, kilo, mal, macro, micro,mid, mis, mono, multi, non, octo, omni, over, para, penta, post, poly, pre, pro, retro, re, semi, sym, sub, super, syn, tele, tetra, therm, trans, tri, un, uni

Table 1: Suffixes and Prefixes list which are used as orthographic features for pos-tagging

BiLSTM is implemented which incorporates the information from the past and future histories 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. The model is trained through gradient descent and back propagation with constant learning rate.

2.1. Orthographic Features as input to BiLSTM

One hot orthographic feature vector is concatenated with the input word embedding and passed as input to the BILSTM which can be clearly seen from the fig 2.

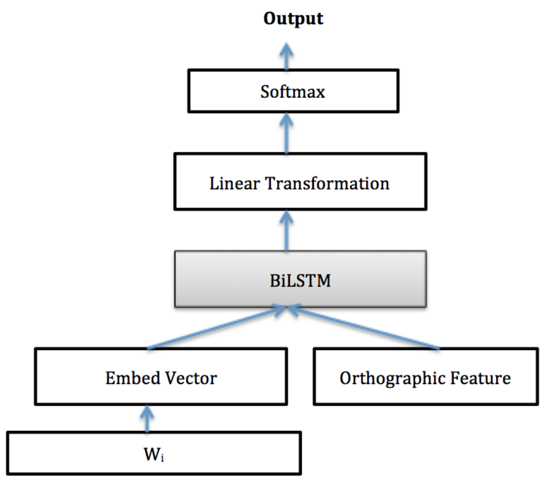


Figure 2: Model 1: BiLSTM model in which orthographic features are concatenated with the input of the BiLSTM.

2.2. Orthographic features as input to Linear Transformation

One hot orthographic feature is concatenated with the output of the BiLSTM which is passed as input to Linear Transformation as shown in fig 3.

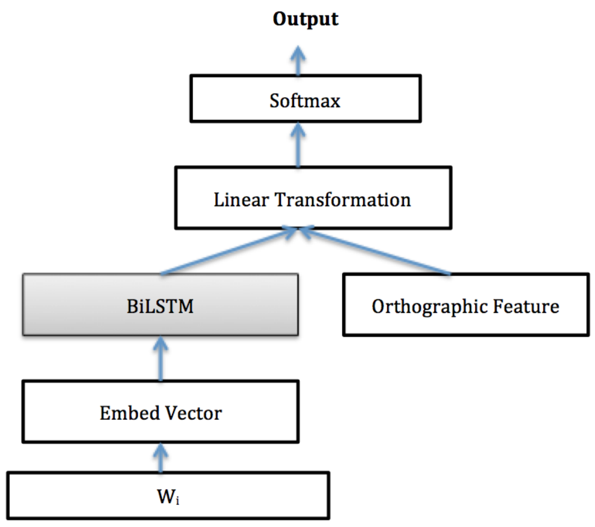
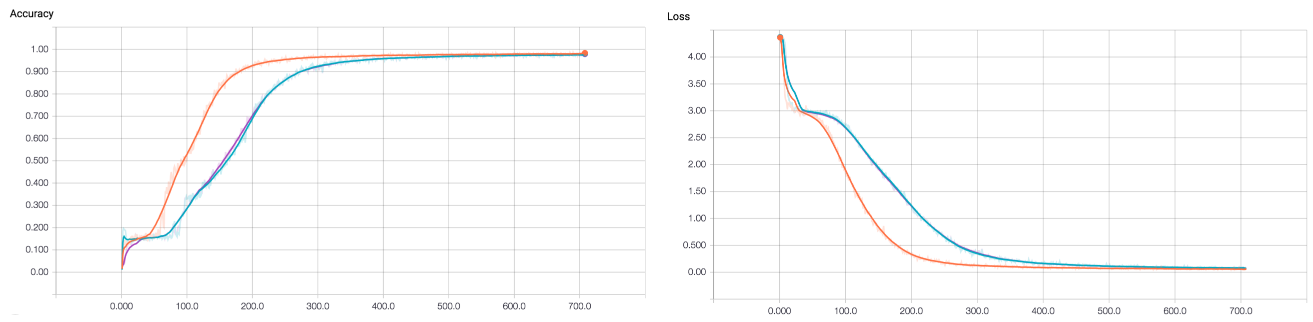


Figure 3: Model 2: BiLSTM model in which orthographic features are concatenated with the output of the BiLSTM.

2.3. Out of Vocabulary Accuracy

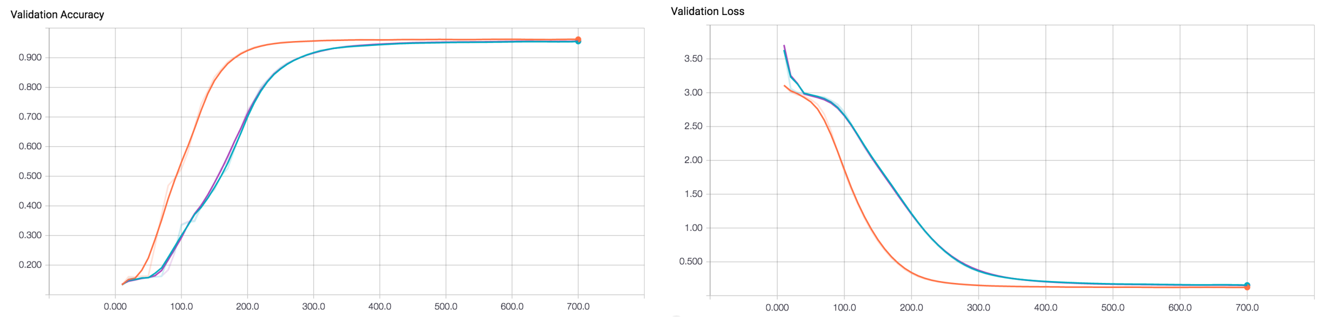
We have also tried to capture the out of vocabulary accuracy so that we can check the orthographic features effect



(a) Training Accuracy

(b) Training Loss

Figure 4: Training Accuracy and Loss for baseline, model 1 and model 2 in which blue curve corresponds to baseline, orange to Model 1 and magenta to Model 2.



(a) Validation Accuracy.

(b) Validation Loss.

Figure 5: Validation Accuracy and Loss for baseline, model 1 and model 2 in which blue curve corresponds to baseline, orange to Model 1 and magenta to Model 2.

on predicting the pos-tag for the out of vocabulary words. To calculate the out of vocabulary accuracy, we created the mask which check which words have id equal to the length of the training set vocabulary size. The words which have these assignment are our Out of Vocabulary words.

3. Experiments

We train and text our model on pos-tagged file Wall Street Journal data from PennTreeBank using 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 plus one UNK symbol for replacing out of vocabulary words.

Baseline system: It is the model in which we are just passing the word as a feature to the BiLSTM, for more reference refer fig 1.

3.1. Accuracy Comparison

Table 2 compares the performance of different model 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 95.7% to 96.3% 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 end with -ing are mostly verb, capitalized words are mostly Proper Noun. And, it can be clearly seen from the OOV accuracy which increases from 57.1% to 71.4%. It shows that the words which model is facing difficult to correctly tag in baseline model; using these orthographic features, it can decide the correct pos-tag to some extent.

Model	**Test Acc (%)**	**Test Loss**	**OOV Acc (%)**	**Train Time (sec)**
Baseline	95.7	0.149	57.1	1917
Model 1	96.3	0.117	71.4	3088
Model 2	95.8	0.144	60.4	2049

Table 2: Comparison of different models with the baseline

From the fig 4 and 5, we can see that training and validation accuracy for the model 1 boost up pretty fast as compared to the model 2 and baseline. Model 1 achieves saturation after 300 iterations, however, other models achieve it after 500 iterations. Model 2 and baseline model have almost the same path which shows model 2 didn't 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 as compared to model 1 and baseline loss (almost same loss for both model). This shows that passing the orthographic features to the BiLSTM input helps in achieving the training and saturation of the model fast as compared to other models.

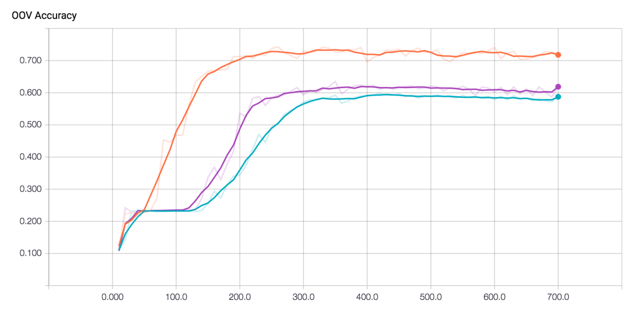


Figure 6: OOV Accuracy for baseline, model 1 and model 2 in which blue curve corresponds to baseline, orange to Model 1 and magenta to Model 2.

From the fig 6, we can see that OOV accuracy for the model 1 boost 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 saturation.

Similarly, from the table 2, we can see that model 1 boost 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 through orthographic features are encoded in BiLSTM from longer 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 longer dependency is not incorporated in model 2 as orthographic features are directly passed to the transformation layer which is using this orthographic features for the current word tagging, no longer dependency exists.

3.2. Training time comparison

In this experiment, we compare the training time (running the model for the same number of epochs) taken by all the models, model 1 take a longer time as shown in table 2. This is obvious because network size increases and it needs longer time to train the model since baseline have less parameters as compared to model 1 and model 2.

Model 1 takes longer time as compared to Model 2 because in model 1, BiLSTM hidden unit size is high (parameters: 300 for word embedding and 116 orthographic features) as compared to Model 2 BiLSTM hidden unit size (containing 300 word hidden unit size). However, their transformation model have same number of parameters. And, we know if network size increases, it needs a longer time to train the model as compared to model having less number of parameters.

However, if we run our model 1 for 5 epochs as we have seen from fig 4 and fig 5 that our model saturates faster as compared to other model; training time for all the model are comparable as show in table 3 without significant loss in testing accuracy.

Model	**Train Time (sec)**
Baseline	1917
Model 1	2186
Model 2	2049

Table 3: Comparison of training time for different models.

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 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 learnt by the BiLSTM model.

To future work, we would like to add other features like morphological features to the BiLSTM input and will use embedding for the orthographical features.