Pramod Chandra Samudrala

Bi-direction LSTM for POS Tagging:

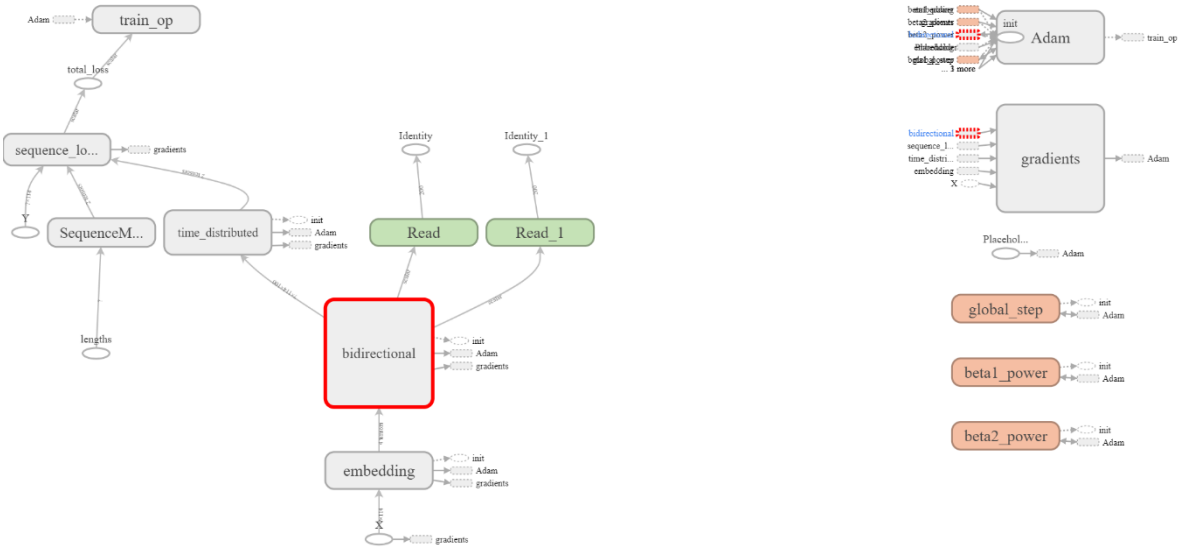

Figure 1:Graph generated by tensorboard

As shown in Figure 1, my model consists of an embedding layer followed by a bidirectional LSTM and then a TimeDistributed Dense layer(fully connected).

Embedding Layer: tf.keras.layers.Embedding is used. L2 regularization is done on embedded weights and weights are initialized using xavier initialization. The parameter here that need to tuned is the embedding size. As we have very less time per language, I used 30 as my embedding size. More the size more variable to train, which will increase computation and take slightly more time, and we need to have more epochs to best fit the embeddings.

Bidirectional-LSTM: As we know that LSTM is better in keep memory of previous time steps than GRU or simple RNN, LSTM was the first choice. An L2 regularizer was added so that we don't overfit training data and can generatlize better. A recurrent dropout and dropout are used.

Regular dropout does dropout at input/outputs whereas recurrent dropout does it at the recurrent units.

More about dropout at https://arxiv.org/pdf/1512.05287.pdf.

Parameter to be tuned is the LSTM recurrent unit size. I used 50 cells.

My reasons for selecting 50 were

1. Low Training Time, so we cant use large no of units
2. More units will tend to overfit the data. As we have very less training data a lower units size is preferred.

3. If we use more units we need to increase the batch size to compensate for speed. Here as we are doing at max 4 epochs, more batch size means less number of updates to weights per epoch, so I observed that learning is a bit slow with high batch size. So had to first tune batch size and then tune the number of recurrent units
4. Observe loss and dev accuracy for different number of recurrent units, which give an intention for number of units to select.

Finally a fully connected layer to map to tag lengths.

A dense layer is wrapped around a time distributed layer. TimeDistributed layer helps in applying the dense layer to all the timesteps, which is required for sequence to sequence mapping.

Learning Rate:

I would say this is main parameter that need to hyper-tuned as far as this assignment is concerned.

I used a learning rate of 0.045 and a decay rule which half's every iteration.