Natural Language Processing
Name:Shashank Mishra
Email: mishras@usc.edu

Part Of Speech Tagging Using RNN.

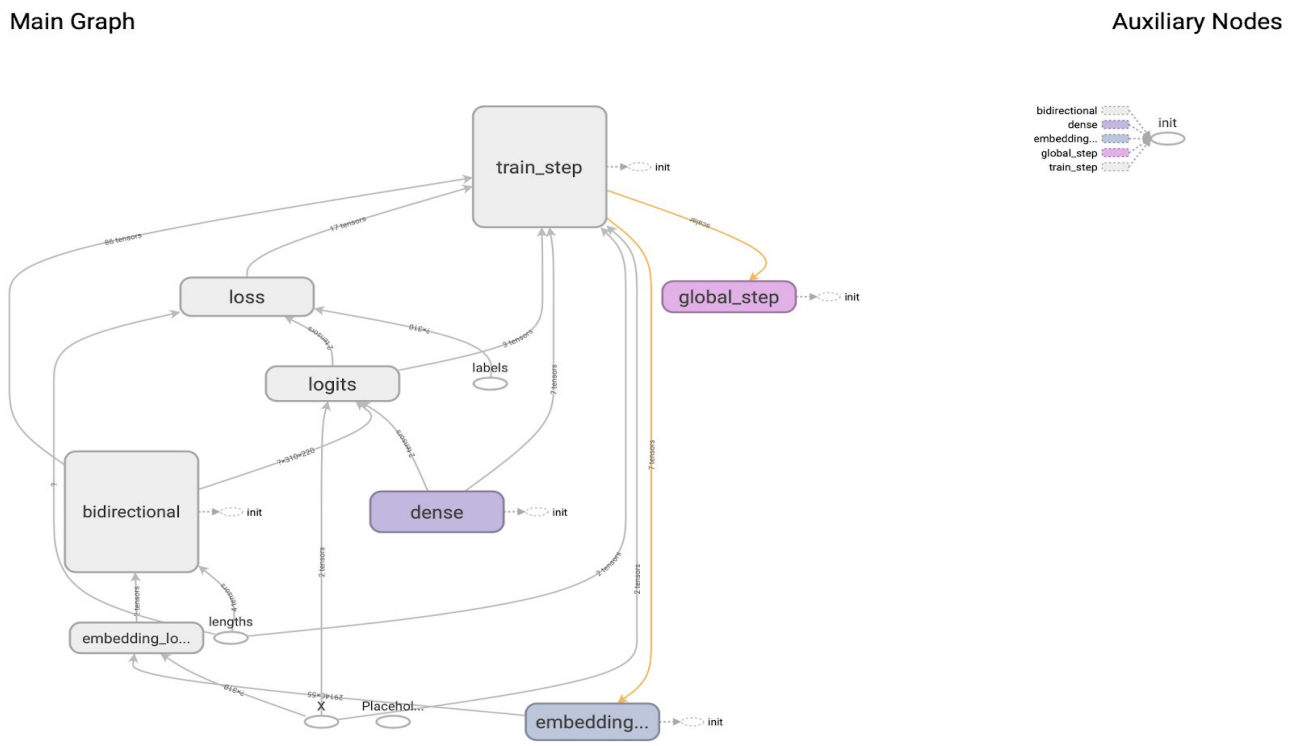


Fig 1. Tensor Graph created by tensorboard to display the graph flow of RNN Model.

Parameters:

S.no	State_size	Embeddings	Batch_size	Learning_rate	RNN
1	**110**	**55**	**32**	**0.005** (0.01/2)	**Bidirectional LSTM**

Accuracy Italian: 95.8 %

Accuracy Japanese: 95.2 %

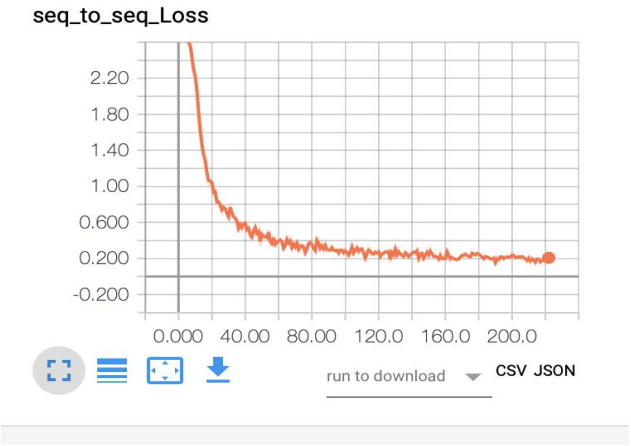
Secret_language: > 95th Percentile

Following are methods used to achieve above given accuracies:

Exploding Gradient:

It occurs When gradient norms become large or NAN in worst case and makes RNN unstable and therefore I clipped the gradient within a threshold of [-1,1] to create a

smooth loss decrease without any spikes.The graph(*fig 2)* shows loss decrease for italian training loss Vs epochs.This also helped in accuracy of secret language tagging.



Gradient Clipping snapshot

opt = tf.train.AdamOptimizer(self.learning_rate)

gradients = opt.compute_gradients(self.loss)

capped_gradients = [(tf.clip_by_value(grad, -1., 1.), var) for grad, var in gradients if grad is not None]

self.train_opt = opt.apply_gradients(capped_gradients)

*Fig 2:*Training loss Vs iterations

Bidirectional RNN

Tensorflow Bidirectional dynamic RNN :

cell = tf.contrib.rnn.BasicLSTMCell(num_units=state_size)

fw,bw=tf.nn.bidirectional_dynamic_rnn(cell_fw,cell_bwl,x_embeddings,sequence_length)

x_embeddings=tf.concat([fw, bw], axis=-1)

Bidirectional RNN is used to stack the 2 independent instances(which is forward and backward) .It gives more power to Simple RNN because, unlike before a particular word is dependent on previous as well as future word in sequence hence we have 2 RNN ,one executes the process in a direction and the second runs the process in the opposite direction.Thus helps in accurate results.

Learning Rate

The Learning Rate is decreased after each epoch as training proceeds so the the gradient loss can converge using formulae

*learning _rate=learning_rate/epoch*2.*

Conclusion:

- Trying various parameters I noticed the power of learning rate and its sensitivity.It is very important to provide an adaptive learning rate with proper decrease after each epochs for proper convergence
- Taking a ratio of 1:2 for embedding and state size works perfect for my model.