☑ checked.

☐ unchecked.

☒ not done.

☑ We have read all the instruction carefully and followed them to our best ability.

☑ We have written the name, roll no in report.

☑ Run sanity_check.sh.

☑ We will be submitting only single submission on behalf of our team.

☑ We have not included unnecessary text, pages, logos in the assignment.

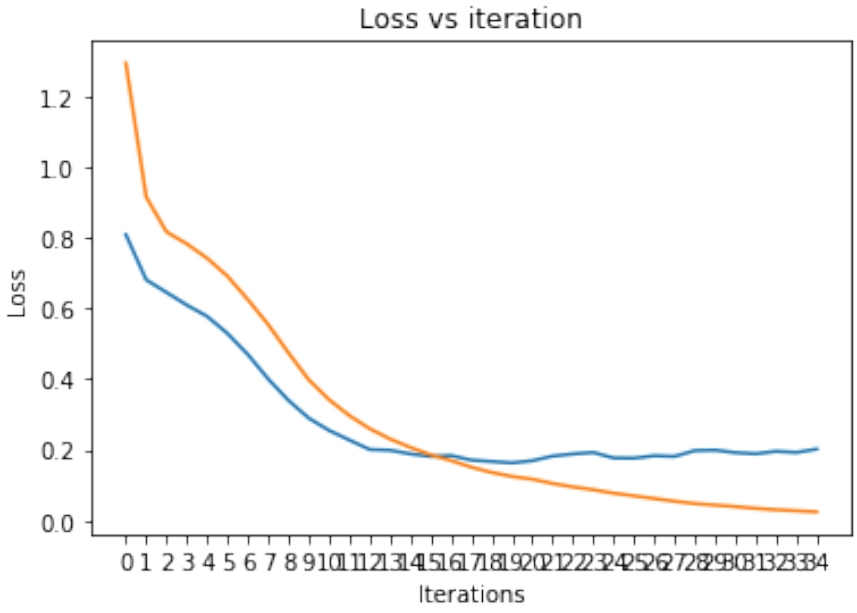
☑ We have not used any high level APIs(Keras, Estimators for e.g.).

☑ We have not copied anything for this assignment.

1. Observations and Inferences.

   (a) A plot of the learning curve showing iterations on the x-axis and negative log likelihood over labels on the y-axis. Make a single plot showing both the training loss and the validation loss.

Solution:



Blue = Validation Loss, Red = Training loss

   (b) Report the parameter setting which gave the best results and the performance on the test data of the model that performs best on the validation data.

Solution:
epochs = 15,
learning rate = .001,

batch size = 128,
Number of Layers for RNN = 2,
Model of RNN = bidirectionl lstm

(c) Report a table with validation accuracies with different hyperparameter settings.

Solution:

Observation table					
MOdel of Rnn	Numbers of layrs	epochs	Learning rate	batch size	validation accuracy
lstm	1	15	0.005	64	0.83
lstm	1	20	0.005	64	0.843
lstm	1	25	0.001	64	0.844
lstm	1	30	0.005	64	0.85
lstm	1	15	.005	32	.81
lstm	1	20	.005	32	.83
lstm	1	25	.001	32	.84
lstm	1	30	.005	3	.86
bdlstm	2	15	0.005	128	0.88
bdlstm	2	20	.001	128	0.87
bdlstm	2	25	.001	128	0.89
bdlstm	2	30	.001	128	0.898
bdlstm	2	15	0.001	64	0.8872
bdlstm	2	20	0.001	128	0.8869
bdlstm	2	25	.001	128	0.888
bdlstm	2	30	0.001	128	0.893

(d) Write down the dimensions of the input and output at each layer

Solution:
ENCODER EMBEDDING INPUT: 47 * 1(BATCHSIZE) OUTPUT: 47 *512 DECODER EMBEDDING INPUT: 87 * 1(BATCHSIZE) OUTPUT: 87 *512

(e) Did you try using only a unidirectional LSTM for the encoder? What was the effect? (including implementation/code for unidirectional and bidirectional LSTM)

Solution:
Yes we used unidirectional and bidirectional LSTM. We know that bidirectional LSTM has access to future information to make current prediction while unidirectional doesn't. This is clearly seen as our Test accuracy increased from 0.2 to 0.3. Bidirectional is able to make better predictions as compared to unidirectional keeping other parameters same.

(f) Explain the attention mechanism you used with equations.

Solution:
For each encoded input from the encoder RNN, the attention mechanism calculates its importance:
importanceij = V * tanh(encodedinputiW1 + DecoderstatejW2)
importanceij is the importance of encoded vector i at decoding step j
W1 , W2 and V are learned parameters

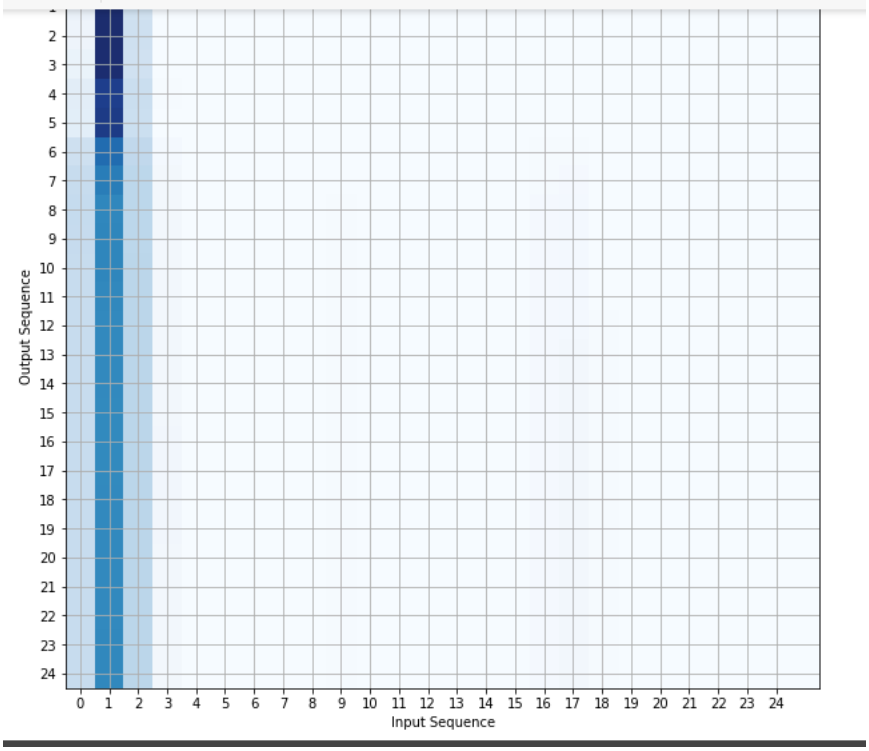
Once we calculate the importance of each encoded vector, we normalize the vectors with softmax and multiply each encoded vector by its weight to obtain a "time dependent" input encoding which is fed to each step of the decoder RNN.
The attention mechanism computes a fixed-size vector that encodes the whole input sequence based on the sequence of all the outputs generated by the encoder

(g) What was the effect of using attention mechanism?

Solution:
Attention mechanism takes into account the input from several time steps, to make one prediction. It distributes attention over the hidden states of several time steps, it accords different degrees of importance, to those inputs.I didn't find any observable increase in accuracy or any other changes.

(h) Plot a visualization of the attention layer weights for a sequence pair. Do you see meaningful character alignments? Do you see one-one, one-many, many-one alignments? Do you see non-contiguous alignments?

Solution:

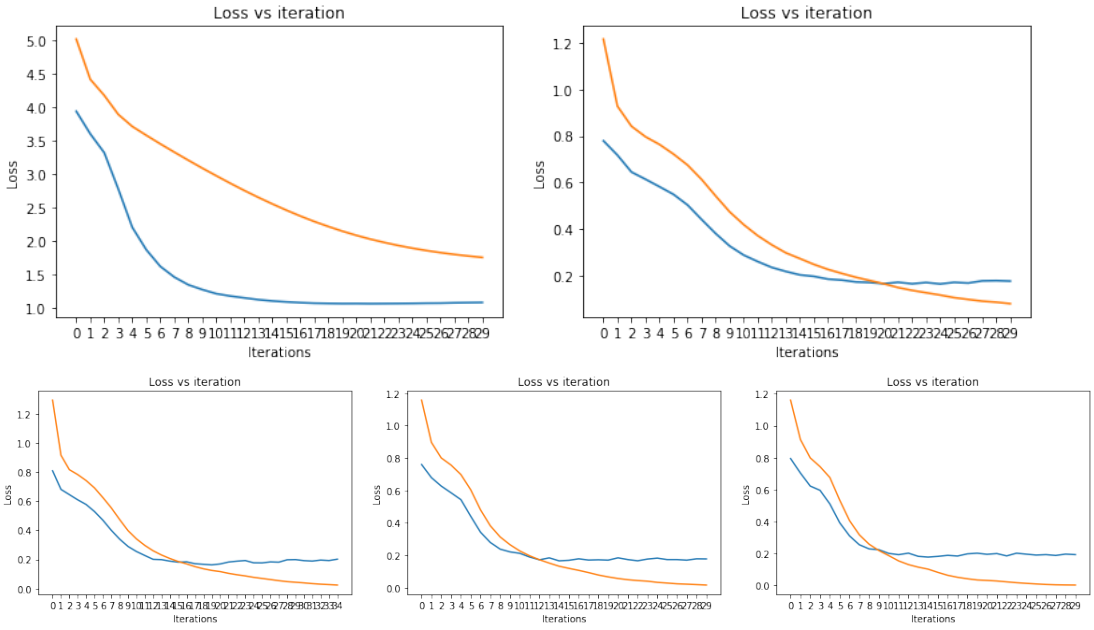


(i) What was the effect of using 2-layered decoder as compared to single decoder?

Solution:
By using 2-layered decoder as compared to single decoder no significant increase in accuracy is observed. Testing accuracy increased by just 2 percent.

(j) What was the effect of using dropout?

Solution:
Blue = Validation Loss, Red = Training loss



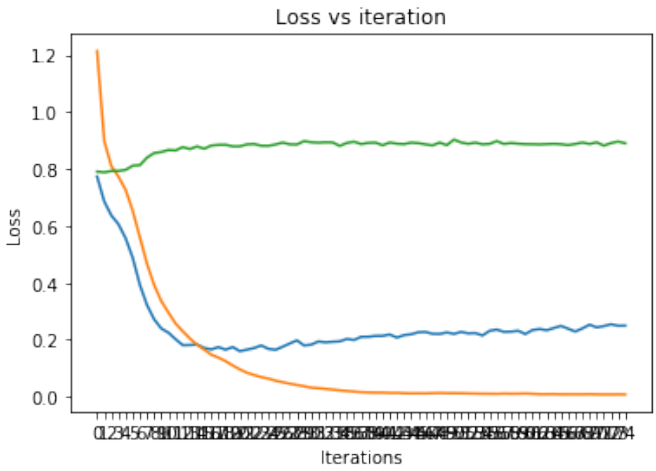
Above plots with keep probability = 0, 0.25, 0.5, 0.75, 1 Keep probability = 0 too much dropout. We can see from above plots if keep probability is zero, then training loss doesn't decrease below validation loss as expected i.e the network underfits and if keep probability is one we can see from output that most of the words are overfit. The network has good accuracy at dropout equal to 0.5.

(k) Is the early stopping criterion optimal? Try training for a few epochs beyond the early stopping epoch. Does the validation loss reduce?

Solution:
No, validation loss doesn't reduce. Early stopping criterion is optimal else overfit occurs in testing outputs.

(l) Instead of using validation loss as early stopping criterion, you can also try using validation accuracy as stopping criterion. How do the two approaches compare?

Solution:
Blue = Validation Loss, Red = Training loss , Green = Validation Accuracy



Yes we can use validation accuracy as early stopping criterion. We can observe from the graph if the accuracy satys constant for certain fixed number of epochs or it increases continuously for the same number of epochs we can stop the training.

(m) If you've implemented beam search, how does beam search compare with greedy decoding?

Solution:
Greedy Decoding : We get a vector 'e' from encoder, use it to generate the target sequence. Feed 'e' to another LSTM cell as hidden state and another vector 'w' as input. The LSTM computes the next hidden state h, then, we apply some function g() so that 's' is a vector of the same size as the vocabulary.Then, apply a softmax to 's' to normalize it into a vector of probabilities 'p'. Now, each entry of 'p' will measure how likely is each word in the vocabulary. Get a corresponding vector 'W' and repeat the procedure. The LSTM will take 'h' as hidden state and 'W' as input and will output a probability vector p1 over next.
Beam Search. Instead of only predicting the token with the best score, we keep track of 'k(beam width) = 5' hypotheses. At each new time step, for these 5 hypotheses we have 'V'new possible tokens. It makes a total of 5V new hypotheses. Then, only keep the 5 best ones, and so on.

(n) What is the effect of beam width on beam search?

Solution:
Beam width, is a parameter which determines how many of the best partial solutions to evaluate. It limits the number to take as input for the decoder. A beam size of 1, only the most probable candidate is chosen as input for the decoder. A beam size of k will decode and evaluate the top k candidates. A larger beam generally means a more accurate prediction at the expense of memory and time.