CS7015 : DEEP LEARNING

PROGRAMMING ASSIGNMENT 4

- seq2seq LSTM with Attention -

April 22, 2018

Student ID:
Namida M - EE15B123
Ganga Meghanath - EE15B025

Contents

1 INTRODUCTION

Weather Gov Data

2 MATHEMATICAL FORMULATION

Write down the mathematical formulation for both Basic Encoder and Hierarchical Encoder. i.e given the Source Table as a collection of Field words f1, f2, ..., fn where each field contains w1f , w2f , ..., wmf tokens. Write the step by steps trans- formations done to reach the text description Y which is basically a sequence of words y1, y2, ..., yk.

2.1 Basic encoder

A single LSTM is used for encoding :

$$h_t^w = LSTM_{encode}^{word}(e_t^w, h_{t-1}^w)$$

The vector output at the ending time is used to represent the entire table :

$$e_T = h_T^w$$

The final state is passed to the decoder.

2.2 Attention mechanism on top of a Basic encoder

A single LSTM is used for encoding :

$$h_t^w = LSTM_{encode}^{word}(e_t^w, h_{t-1}^w)$$

The vector output at the ending time is used to represent the entire table :

$$e_T = h_T^w$$

Implementing Attention over the LSTM,

$$e_{jt} = V_{Att}^T \, tanh(U_{Att} \, h_j^w + W_{Att} \, s_t + b_{Att})$$

$$\alpha_{jt} = softmax(e_{jt})$$

$$c_t = \sum_{j=1}^{T} \alpha_{jt} h_j^w$$

The final state is passed to the decoder as $[e_T, c_t]$.

2.3 Hierarchical encoder

A layer of LSTM is placed on top of the words in a sentence :

$$h_t^w = LSTM_{encode}^{word}(e_t^w, h_{t-1}^w)$$

The vector output at the ending time is used to represent the entire sentence :

$$e_s = h_T^w$$

Another layer of LSTM is placed on top of all sentences :

$$s_t^s = LSTM_{encode}^{sentence}(e_t^s, s_{t-1}^s)$$

Hence, the entire document(table) is represented by :

$$e_{table} = s_T^s$$

The final state is passed to the decoder.

2.4 Attention mechanism on top of a Hierarchical encoder

A layer of LSTM is placed on top of the words in a sentence :

$$h_t^w = LSTM_{encode}^{word}(e_t^w, h_{t-1}^w)$$

The vector output at the ending time is used to represent the entire sentence :

$$e_s = h_T^w$$

Implementing Attention over the LSTM over words,

$$e^w_{jt} = (V^w_{Att})^T \ tanh(U^w_{Att}\ h^w_j + W^w_{Att}\ h^s_t + b^w_{Att})$$

$$\alpha^w_{jt} = softmax(e^w_{jt})$$

$$c^w_t = \sum_{j=1}^{T} \alpha^w_{jt} h^w_j$$

Another layer of LSTM is placed on top of all sentences :

$$h^s_t = LSTM^{sentence}_{encode}([e^s_t, c^w_t], h^s_{t-1})$$

Hence, the entire document(table) is represented by :

$$e_{table} = s^s_T$$

Implementing Attention over the LSTM over sentences,

$$e_{jt} = V^T_{Att}\ tanh(U_{Att}\ h^s_j + W_{Att}\ s_t + b_{Att})$$

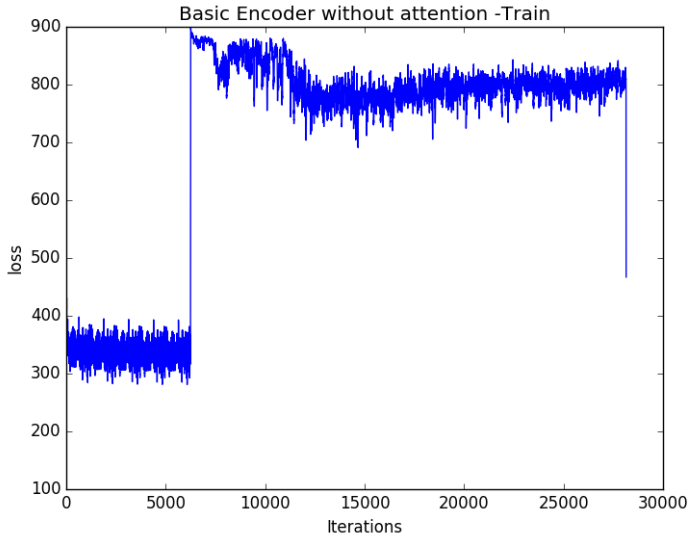
$$\alpha_{jt} = softmax(e_{jt})$$

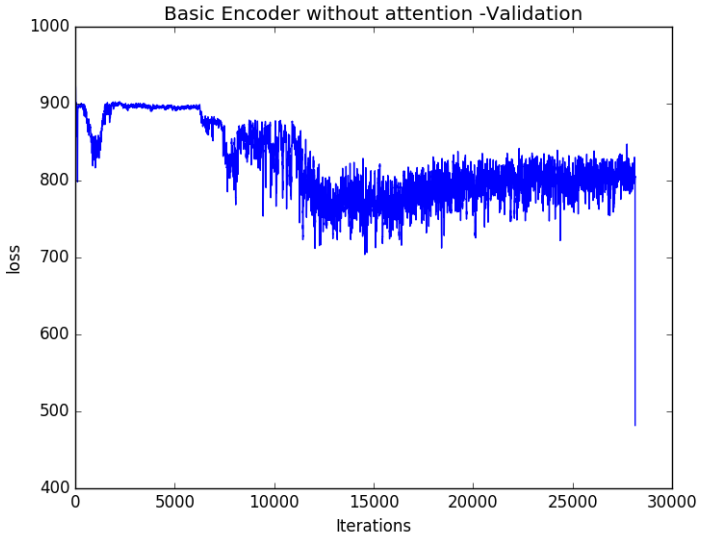
$$c_t = \sum_{j=1}^{T} \alpha_{jt} h^s_j$$

The final state is passed to the decoder as $[e_{table}, c_t]$.

3 PLOTS OF TRAINING AND VALIDATION LOSS

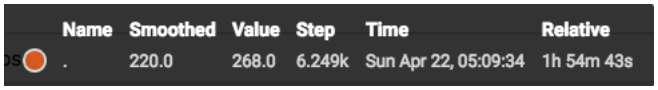
3.1 Basic encoder-decoder



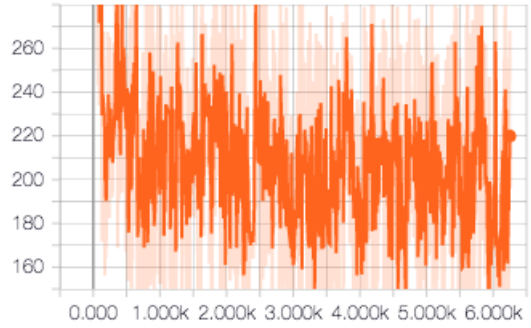


3.2 Basic encoder-decoder with attention mechanism

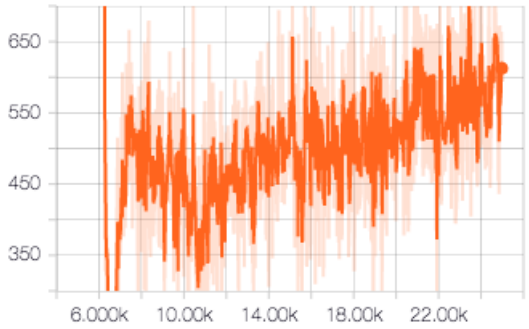
The tensorflow scalars generated for training is as follows :
For a learning rate of 0.0001 and Adam optimizer



Name	Smoothed	Value	Step	Time	Relative
.	220.0	268.0	6.249k	Sun Apr 22, 05:09:34	1h 54m 43s

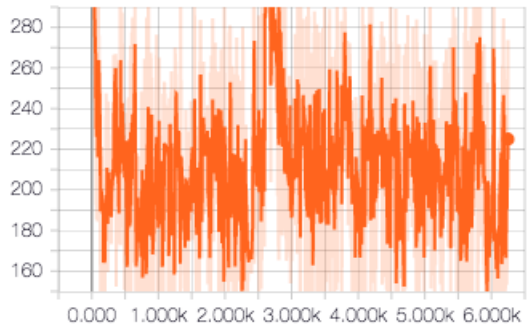


Name	Smoothed	Value	Step	Time	Relative
.	612.8	620.1	24.98k	Sun Apr 22, 15:01:48	9h 52m 2s

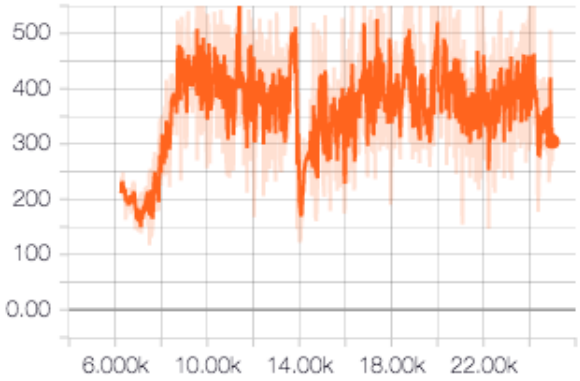


For a learning rate of 0.001 and Adam optimizer

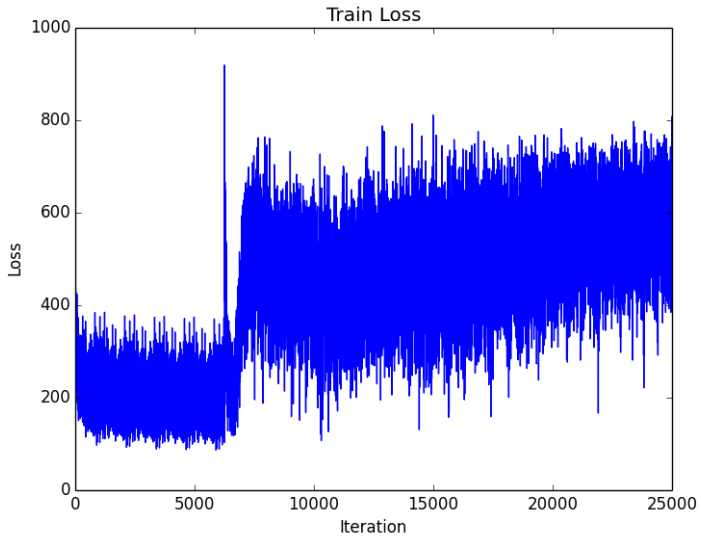
Name	Smoothed	Value	Step	Time	Relative
.	225.1	273.7	6.249k	Sun Apr 22, 06:01:13	1h 55m 21s

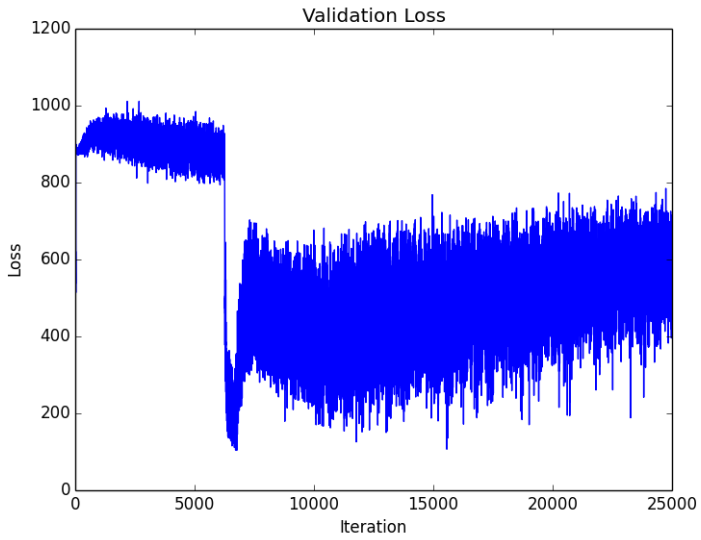


Name	Smoothed	Value	Step	Time	Relative
.	329.2	337.6	24.97k	Sun Apr 22, 15:52:34	9h 51m 5s

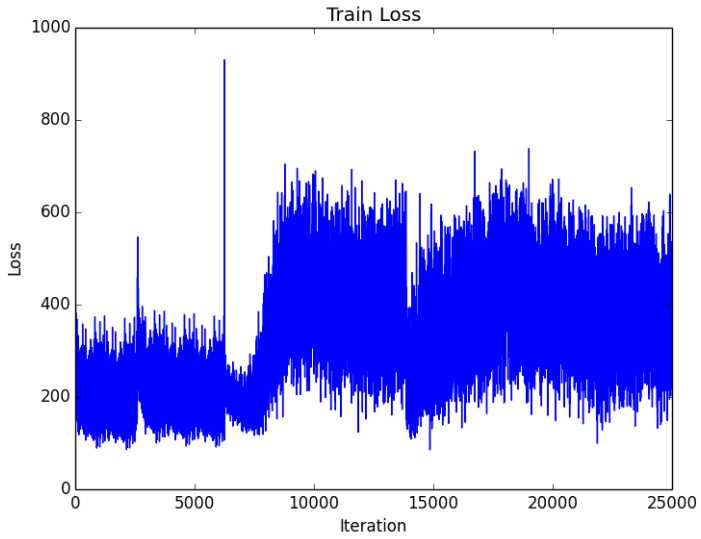


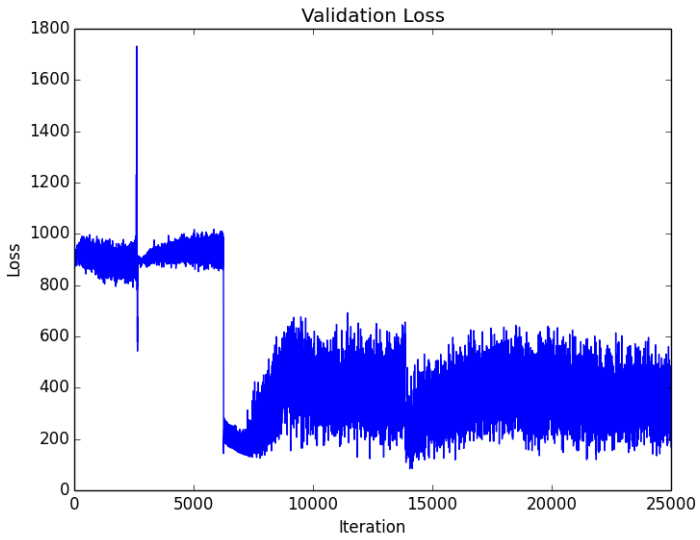
The training and validation loss plotted in python has been shown below :
For a learning rate of 0.0001 and Adam optimizer





For a learning rate of 0.001 and Adam optimizer





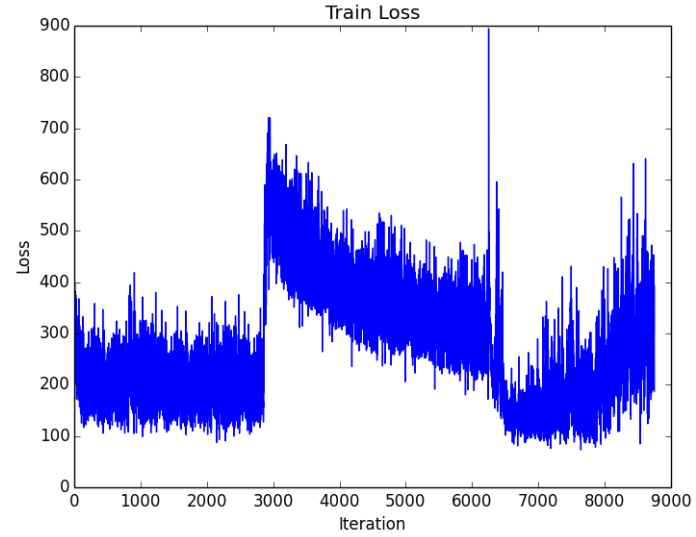
4 MODELS

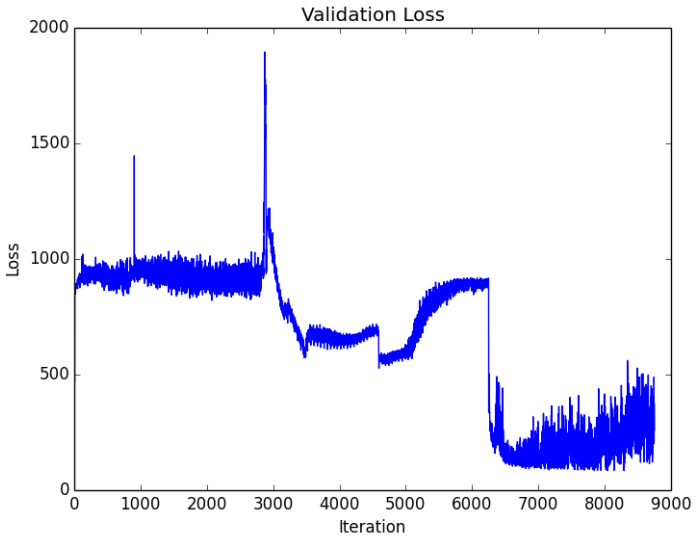
4.1 Best Model

The best model was found to be using Basic encoder with Attention. The training was stopped when the validation error was minimum, and this was found out through early stopping and the model with the least validation score was restored.

4.1.1 Parameter setting

For a learning rate of 0.001 and Adam optimizer, the algorithm was run for 8750 iterations (1250 iterations = 1 epoch)





4.1.2 Performance

The model gave a validation score of 295.8670654296875 for the entire validation set at the end of training.

4.2 Model 1 : Basic encoder-decoder

Configuration details :

Layer	Input Shape	Output Shape
INEMBED	(20 × max_index(batch))	((20 × max_index(batch) x 100)
ENCODER	((20 × max_index(batch) x 100)	(20 × 512)
DECODER	(20 x 150 x 512)	(20 x 150 x 512)
SOFTMAX	(20 x 150 x 512)	(20 x 150 x 512)
OUTEMBED	(20 x 150 x 512)	(20 x 150 x 512)

The encoder is a bidirectional RNN with forward and backward cell size as 512. So, when passing the final state to the decoder , we have to concatenate the states. We have chosen to project the states to the decoder.
So, the output from the encoder is an LSTM state tuple of the form $(c_{fw}, h_{fw}), (c_{bw}, h_{bw})$.

SOME TEST SUMMARIES

- partly sunny , with a high near 80 . breezy , with a south wind between 11 and 21 mph .
- showers , with thunderstorms also possible after 10pm . low around 38 . south wind wind around and mph mph . chance of precipitation is is % . .

4.3 Model 3 : Basic encoder-decoder with Attention mechanism

Configuration details :

Layer	Input Shape	Output Shape
INEMBED	(20 × max_index(batch))	((20 × max_index(batch) x 100)
ENCODER	((20 × max_index(batch) x 100)	(20 × 512)
DECODER	(20 x 150 x 512)	(20 x 150 x 512)
SOFTMAX	(20 x 150 x 512)	(20 x 150 x 512)
OUTEMBED	(20 x 150 x 512)	(20 x 150 x 512)

The dimensions remain the same as in the case of a basic decoder without attention. Here, instead of passing the final state, we pass a weighted sum of the weights, over the time steps.

SOME TEST SUMMARIES

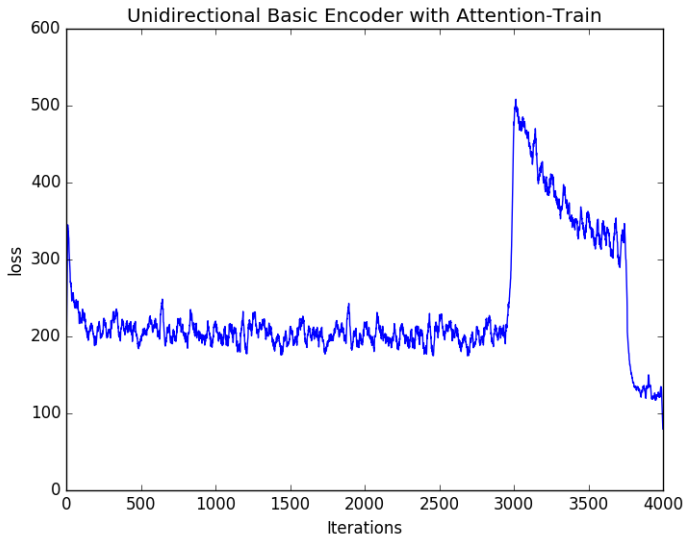
- A 20 percent chance of showers . mostly cloudy , with a with a near . . wind wind wind and with with mph as high as
- Mostly cloudy , with a low around . . south wind between 5 and 15 mph .

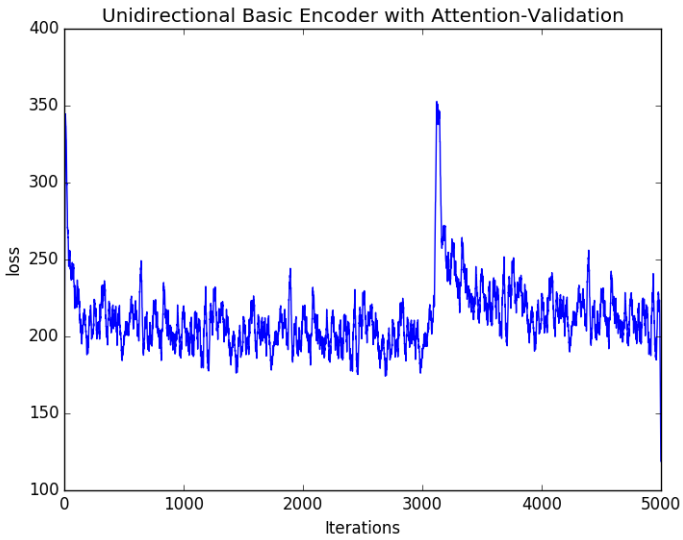
5 ANALYSIS

5.1 Effect of using unidirectional LSTM for the encoder

The performance of the encoder-decoder model using a uni-directional LSTM is not as good as in the case of a bidirectional LSTM, as the state only captures past context, not (future context).

The learning curves are as follows:





SOME TEST SUMMARIES

- Increasing clouds , with a low around . . wind wind between and wind and mph mph .
- A chance of rain before before . cloudy . cloudy . cloudy , a , a , a , a . . . . . .

5.2 Basic vs Hierarchical Encoder (Without Attention)

The Hierarchical encoder should ideally perform better, as the basic encoder without attention, doesnt know which part of the table is important, while the Hierarchical encoder by structure is made to pay attention to the respective part of table.

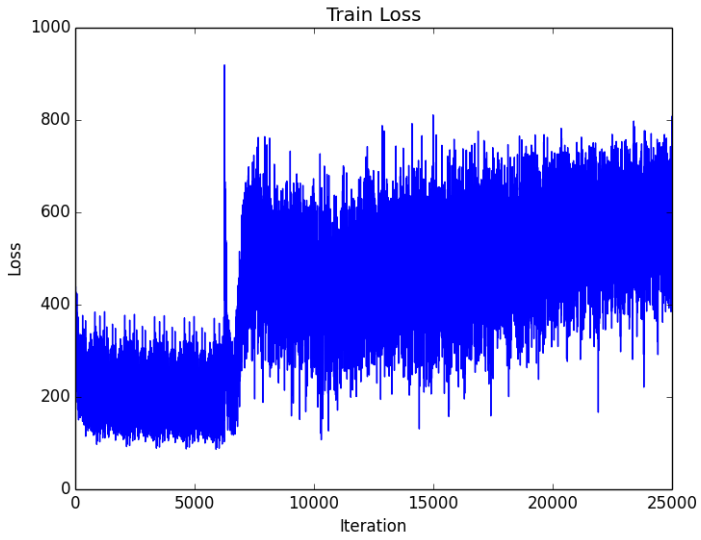
5.3 Effect of using Attention mechanism

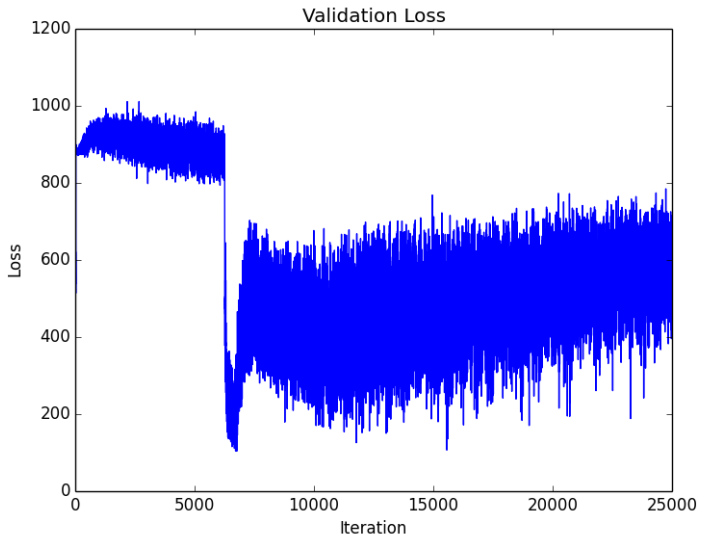
Introducing attention mechanism to the model causes an improvement in performance. Naturally, when summarizing a table we tend to focus on different aspects of the data as we write our summmary. In case of a basic encoder, the model has to learn how each row corresponds to an attribute, and within each row ,it needs to learn to focus as different aspects of that attribute.
For the hierarchical encoder, the model will learn how to pay attention to different parts of a row(on the first level), and then, over the rows.

5.4 Optimality of early stopping

On running for 20 epochs, the results were obtained as follows :

For a learning rate of 0.0001 and Adam optimizer

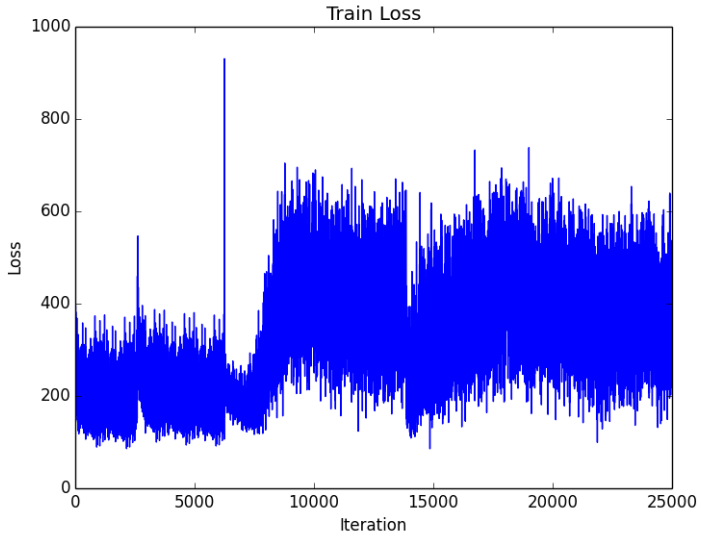


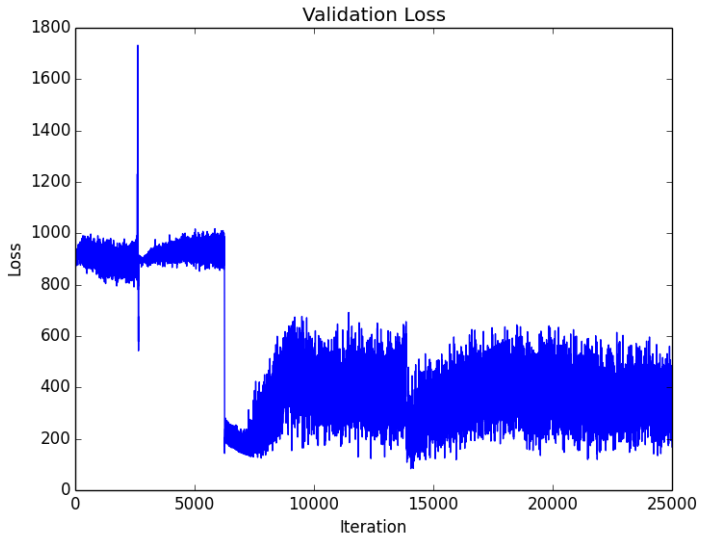


In this case, on an average, the validation loss doesn't look like it's going to decrease after running it for epochs beyond the early stopping criterion. We can see that, on an average, the loss seems to be increasing. But since we use the model that gives the lowest validation loss, we see that after the first minima, the graph on an average goes up and then the loss decreases again to around the minimal value and then seems to increase. At the same time,

train loss also increases. Hence, had we run it for a lot more epochs, there may come a point which is more optimal. LSTMs have a large number of parameters that need to be trained. Hence we cannot state anything solid.

For a learning rate of 0.001 and Adam optimizer

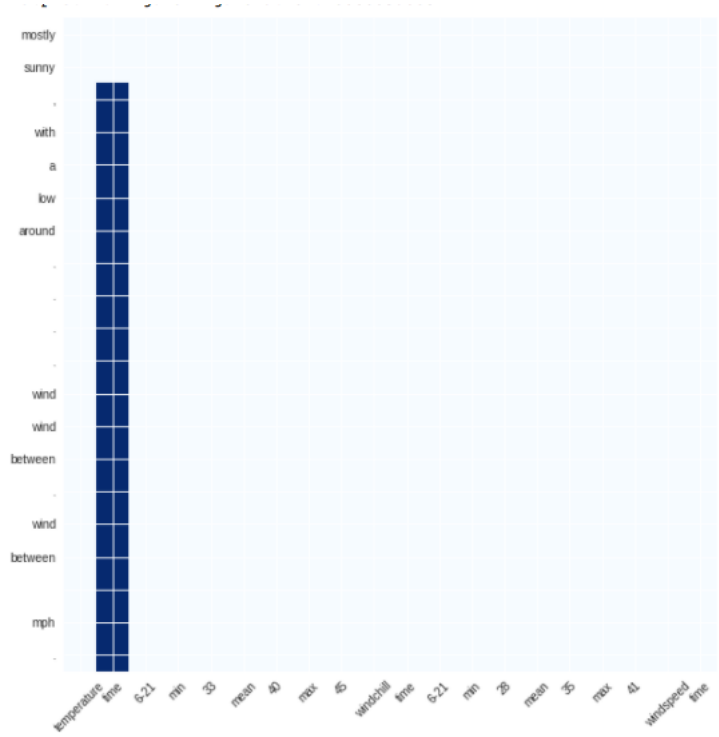




Here both the training and validation loss seems to be decreasing after a couple of epochs from the stopping criterion used from early stopping. Hence had we continued training, we might have been able to find a pint of minima which is more optimal that the one given by the early stopping criterion.

Google colab allocates a GPU for 12 hours and that is enough only for 20 epochs and hence we didn't get a chance to explore further. Although we did try downloading the saved model, it gave a timeout error.

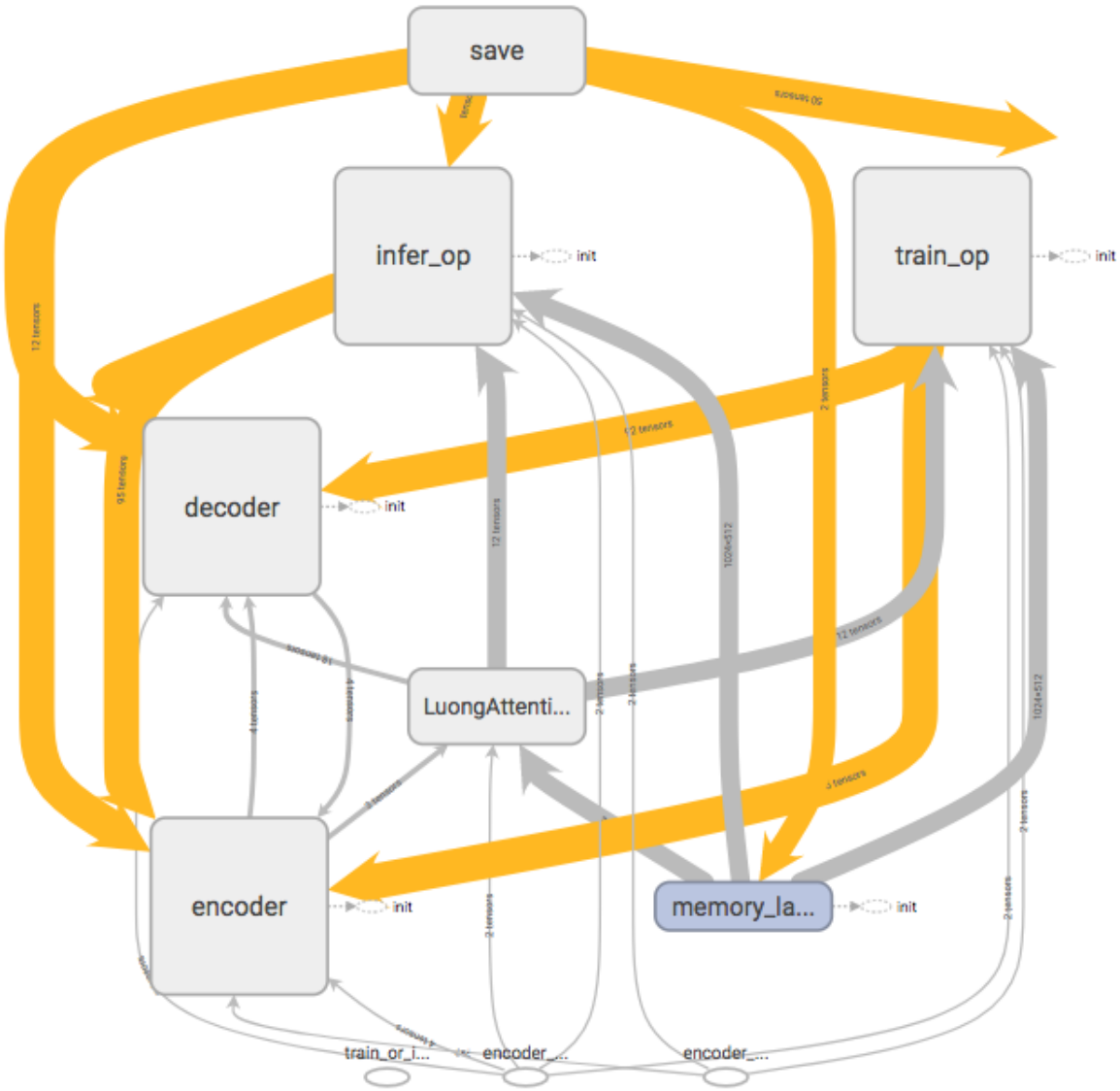
5.5 Visualization of Attention layer weights for a sequence pair



Unable to train for meaningful model due to time constraints, and runtime disconnect on google colab

6 APPENDIX

6.1 Main Graph


save
infer_op
init
train_op
init
decoder
init
LuongAttenti...
encoder
init
memory_la...
init
train_or_i...
encoder_...
encoder_...

Bibliography

[1] Mitesh M Khapra. *CS7015 Deep Learning: Lecture 13-18*, Indian Institute of Technology Madras, 2018