Assignment 4

Your name: Prateek Gupta
Your GTID:903644658

Seq2Seq Results

Used Manual Seed = 0 for comparison

Results for default configuration using RNN	
Training Loss	4.5949
Training Perplexity	98.9749
Validation Loss	4.5227
Validation Perplexity	92.0864

Results for default Configuration Using LSTM	
Training Loss	4.1856
Training Perplexity	65.7338
Validation Loss	4.0468
Validation Perplexity	57.2164

Result for your Best Model using RNN after hyperparameter tuning	
Training Loss	4.5350
Training Perplexity	93.2190
Validation Loss	4.4382
Validation Perplexity	84.6255

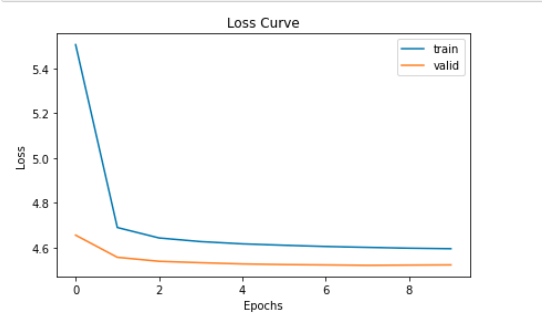
Resut for your Best Model using LSTM after hyperparameter tuning	
Training Loss	3.7131
Training Perplexity	40.9791
Validation Loss	3.7847
Validation Perplexity	44.0205

Your best model configuration for RNN after hyperparameter tuning
encoder_emb_size = 128 encoder_hidden_size = 256 encoder_dropout = 0.9 decoder_emb_size = 128 decoder_hidden_size = 256 decoder_dropout = 0.5 learning_rate = 1e-3 EPOCHS = 10

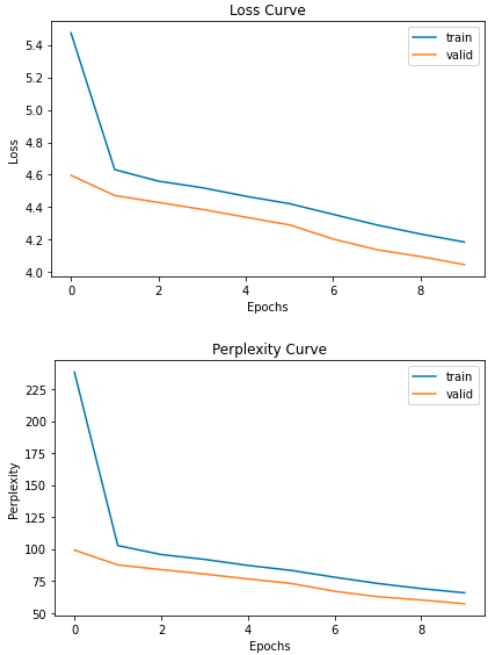
Your best model configuration for LSTM after hyperparameter tuning
encoder_emb_size = 32 encoder_hidden_size = 2048 encoder_dropout = 0.5 decoder_emb_size = 32 decoder_hidden_size = 2048 decoder_dropout = 0.5 learning_rate = 1e-3 EPOCHS = 10

Seq2Seq Curves

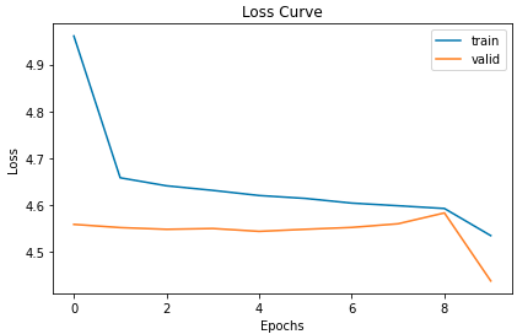
Put the plots for loss/perplexity curves (training & validation) for your configuration with default setting and for your best model here.

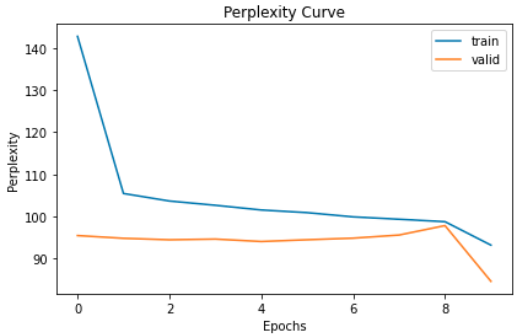


Default RNN

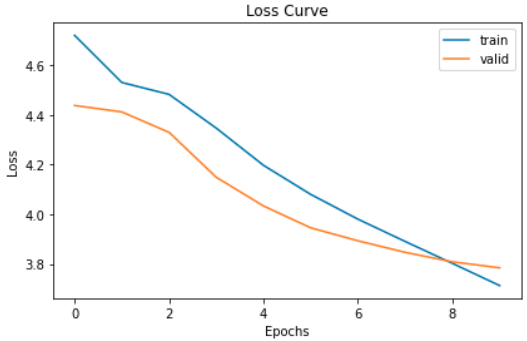


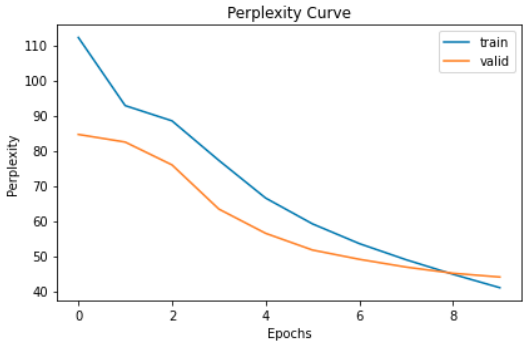
Default LSTM





Best RNN





Best LSTM

LSTM performance is better than RNN and is our best Seq2Seq Model (excluding transformer)

Seq2Seq Explanation

Explain what you did here and why you did it to improve your model performance. You can use another slide if needed.

From the initial chart of RNN, we see that the model took a steep fall from epoch 0 and then the training loss stopped moving. On closer inspection of the gradients passed, we found that the model experienced a gradient explosion because number of timesteps which was clipped in the code. After the steep fall, the loss then move to a region of local minima/saddle point. To understand this phenomema, we increased the learning rate a little and found the loss to diverge from epoch 1 itself on the training data. To fix this, we tried tweaking the asymmetrical drop out parameter so that the randomness can alleviate some of the gradient issues we experienced. We were able to improve the learning process, however the qualititative assessment of prediction showed the model relied on simply returning the most common word in the vocabulary 'a' which reduced the loss and perplexity. Further, we tried changing the min_freq in spacy and found the raw[0:9] contains low frequency words like 'burglers'. So, we increased the overall capacity of the model to memorize some of these words. The model suffers from extreme overfitting probably due to lack of pretrained embedding and a bad initialization. By all these changes, we can see that in the final tuned RNN learning for loss and perplexity, the model loss increases and then decreases at the end. We believe RNN is struggling with lots of local minima and saddle point. We did not increase the epoch and learning rate as the model is experiencing strong overfitting on most common words and is not learning the language relationships.

From the initial chart of LSTM, we see that the model took a steep fall from epoch 0 but it is not as bad as rnn curve. So, we believe, the initialization is better and we also observe the loss is slowly decreasing. However, qualitatively on checking the prediction, we see the model is again focussing on high frequency words. Lack of pretrained embedding and a learnable embedding exacerbate the problem. The model has more parameters and need to learn about controlling gates at the same not overfitting on high frequency words. To mitigate this, we first tried to reduce embedding layer so the model has lesser burden of finding similar words. Next, we try to increase the hidden representation of LSTM layer. This is done to simply memorize the training relationship. However, we again find the high frequency words are part of prediction. This could be possible because the cell gates do not work as intended and are always open. So, we try to solve the overfitting problem and not minimize the loss function/perplexity by looking at the output translations. We reduce dropout rate to 0.5. The model struggles with low frequency words which can be seen from the translation. Additionally, from chart, we see that at epoch 8, our model perform better on training data. We tried to find if there is multi modality and the training curves cross again when the model learns the relationship better. Unfortunately, that was not the cae. We trained till epoch 100 (not shown) and found the model simply translates everything to 'a' after the second word and continues to overfit on common words. The final LSTM model is selected with a mixture of qualitative assessment (translation shown at the end of ppt) and quatitative performance.

Transformer Results **Used Manual Seed = 0 for comparison**

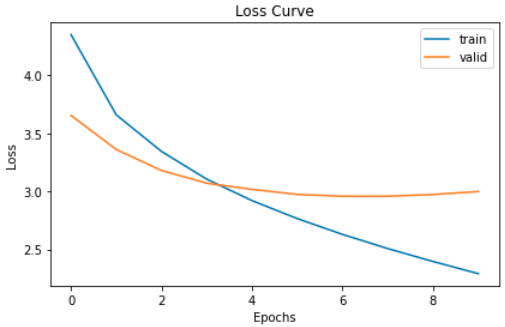
Put your results from training before and after hyperparameter tuning here.

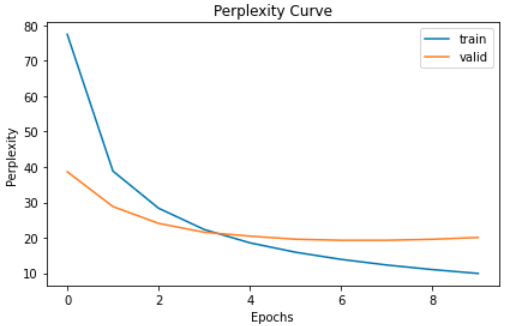
Results for default configuration			
Training Loss	2.2920	Validation Loss	2.9990
Training Perplexity	9.8945	Validation Perplexity	20.0652
Result for your Best Model			
Training Loss	2.6669	Validation Loss	2.9356
Training Perplexity	14.3956	Validation Perplexity	18.8329
Your best model configuration after hyperparameter tuning			
learning_rate = 3e-4 EPOCHS = 15 MAX_LEN = 15 SRC.build_vocab(train_data, min_freq = 2,max_size=80000) TRG.build_vocab(train_data, min_freq = 2,max_size=80000)			

Table 2

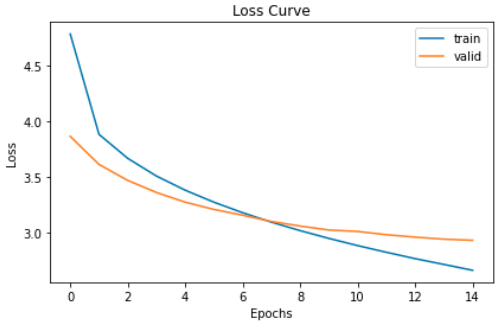
Transformer Curves

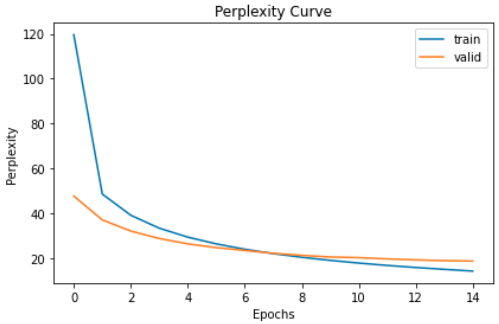
Put the plots for loss/perplexity curves (training & validation) for your configuration with default setting and for your best model here.





Default Transformer





Best Transformer

Transformer Explanation

Explain what you did here and why you did it to improve your model performance. You can use another slide if needed.

From the initial chart of transformer, we see that the model overfits very quickly around epoch 3-4. To alleviate this problem, we simply reduced the learning rate. This was done in two ways – ReduceLROnPlateau and StepWise reduction. Based on results, we kept the former optimizer. The learning rate was reduced till 1e-6. However, we again notice that the model start translating everything to 'a'. So, we increased the learning rate to 3e-4 and increased the epochs to 15. We did not concentrate on drop out and changing the number of heads as it would require substantial redoing the model architectural design all over again. Next, we notice that the model translations improve with the perplexity curve. For example, translation at epoch 5 and translation at epoch 8 are different. Even with overfitting that can be seen from the curve after epoch 7 in the curve, we see that the quality of translation improves. This could be because of the randomness that we started with embedding layer. The words may be incorrect but they are closer to synonyms. May be starting with a pretrained embedding layer and a non trainable positional encoding might address the issue. We also tried reducing the Max len of the sentence. As the translation resorts to repeating the word or simply predicting 'a', we tried to reduce the burden of focussing all words. Similarly, for vocab, we put a max vocab limit of 80000. This improved the quality of translation a little. By making these changes, we were able to improve the performance on the validation data – 18 perplexity and incrementally improve the quality of translation.

Transformer Translation Results

Put translation results for your best model (1st 9 sentences) here

Input sentence	Back translation
'<sos>', 'a', 'man', 'cooking', 'burgers', 'on', 'a', 'black', 'grill', '.', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'man', 'is', 'a', 'on', 'on', 'black', 'grill', '.', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'a', 'man', 'and', 'woman', 'fishing', 'at', 'the', 'beach', '.', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'man', 'and', 'a', 'woman', 'a', 'on', 'on', 'the', '.', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'a', 'man', 'in', 'a', 'harness', 'climbing', 'a', 'rock', 'wall', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'man', 'in', 'a', 'a', 'is', 'climbing', 'a', 'a', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'a', 'cute', 'baby', 'is', 'smiling', 'at', 'another', 'child', '.', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'baby', 'baby', 'smiles', 'a', 'another', 'child', '.', '.', '.', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'a', 'female', 'playing', 'a', 'song', 'on', 'her', 'violin', '.', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'woman', 'playing', 'a', 'guitar', 'on', 'a', '.', '.', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'a', 'person', 'on', 'a', 'snowmobile', 'in', 'mid', 'jump', '.', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'person', 'on', 'a', 'a', 'mid', 'in', 'in', '.', '.', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'three', 'men', 'competing', 'in', 'a', 'hurdle',	'<sos>', 'three', 'men', 'are', 'in', 'a', 'a', 'a', '.', '.',

LSTM Translation Results

Put translation results for your best model (1st 9 sentences) here

Input sentence	Back translation
'<sos>', 'a', 'man', 'cooking', 'burgers', 'on', 'a', 'black', 'grill', '.', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'man', 'is', 'a', 'a', 'on', 'a', '.', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'a', 'man', 'and', 'woman', 'fishing', 'at', 'the', 'beach', '.', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'man', 'and', 'a', 'woman', 'are', 'a', 'the', '.', '.', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'a', 'man', 'in', 'a', 'harness', 'climbing', 'a', 'rock', 'wall', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'man', 'in', 'a', 'is', 'is', 'a', 'a', 'a', '.', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'a', 'cute', 'baby', 'is', 'smiling', 'at', 'another', 'child', '.', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'young', 'girl', 'is', 'a', 'a', '.', '.', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'a', 'female', 'playing', 'a', 'song', 'on', 'her', 'violin', '.', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'woman', 'is', 'a', 'a', 'on', 'on', '.', '.', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'a', 'person', 'on', 'a', 'snowmobile', 'in', 'mid', 'jump', '.', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'person', 'on', 'a', 'bike', 'on', 'a', 'a', 'a', '.', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'three', 'men', 'competing', 'in', 'a', 'hurdle',	'<sos>', 'three', 'men', 'are', 'in', 'a', 'a', 'a', 'a', '.',

Compare LSTM to Transformer

Compare your LSTM results to your Transformer Results both quantitatively and qualitatively and explain the differences.

From the translation, we see that the transformer takes advantage of self-attention mechanism even with the starting point of random embedding layers. For instance, for the first translation, the LSTM model is confused after the second word and ends up repeating itself or the most common word in the vocabulary - "a". Additionally, the transformer trains faster due to remove of recurrence and complete reliance on attention. Even though, we have dedicated gates in LSTM for remembering long term dependencies, the hidden state passed from encoding is not enough to generate coherent sentences. This could be seen again from translation number 4. However, we see that LSTM is able to infer that snowmobile is similar to a bike which the transformer did not catch. We believe that this happened because tranformer has to learn two sets of embedding while LSTM has to learn only one embedding during encoding. Transformer is able to learn actions possibly because of positional encoding.

Quantitatively, we see from the final loss and perplexity numbers, that the transformer performs better and trains faster. From the table, 18 is the validation perplexity for transformer, while 44 is the validation perplexity for transformer. 2.93 is the validation loss of transformer while 3.71 is the validation loss for LSTM. The drop in performance in LSTM could be due to long term dependencies. Subjectively, we see 4 out of 9 translations are better for transformer. 1 translation is better in LSTM.