Skip Connections and Multiple Matrices in Recurrent Neural Networks

Mihir Mongia

Abstract

A problem in inferring functions related to sequences is that it is not easy to train recurrent neural networks to capture long term dependencies. Several partial solutions exist to circumvent the problem such as LSTM(Long Term Short Term Memory)s and GRU(Gated Recurrent Neural Networks)s but the jury is still out on what is optimal for real world applications. We experiment with the use of skip connections similar to Residual Nets. We show that using basic RNNs we can match the performance of LSTMs. The advantage of this is that LSTMS require much more computation both in training and testing. We also use different matrices to see if we can improve performance on language tasks.We however are not able to achieve state of the art on the Stanford Sentiment Treebank dataset using multiple matrices and/or skip connections.

Introduction

Solving the long term dependency problem is an important issue. One can think of applications such as predicting DNA sequences where it is possible elements of a sequence 100 apart could affect the output. Even in language this is a very important problem. A movie review may have good things to say about a movie and bad things to say about a movie over the course of a large paragraph. A neural network needs to be able to be able automatically capture whether it should focus on the long term or short term and it needs to have the capabilities to capture long term relationships. We know that because of the vanishing gradient problem learning long term relationships in basic RNNs is difficult. The common weapon to tackle long term dependencies are LSTMs. LSTMs require a lot of computation but are generally used for most sequential tasks.

At the same time we also are trying to show that using multiple matrices can be advantageous in processing natural language tasks. Quite often any RNN will use the same W matrix for each time step because we want to process all time steps equally. This however does not seem to capture the fact that there can be more complexity in language than just using one matrix. There has not been much work on how using multiple matrices might or might not help in natural language processing. We explore experimentally to see if there are any advantages.

Overall we try to show that skip connections paired with basic RNNs might be a better alternative to LSTMs for sequential tasks. In addition, we aim to test the hypothesis that using multiple matrices might help in natural language tasks. To test the power of skip connections,we first set up a synthetic task that requires realizing long term dependencies and see how adding skip connections and using multiple matrices helps in performance. We then port over skip connections and multiple matrices to natural language tasks. We see how we perform on the Stanford Sentiment Treebank Dataset.

Related Work

Delay networks that give feedback from the output back into the input were first proposed by Lin et al 1996. This followed from the idea of incorporating delays in feedforward neural networks (Lang and Hinton, 1988). These type of skip connections however give delays to signal flowing on the skip connection. In a paper by Microsoft Research Asia "Deep Residual Learning for Image Recognition", in order to avoid the vanishing gradient problem, the authors add skip connection skipping every 2 layers to their deep convolutional neural network. In this way the gradient from the top layer will definitely reach the bottom layer. They are able to report better generalization error. This as far as we know has not been applied to basic RNNs and it is possible that the improved generalization error seen in Residual Nets could also could be paralleled with improved generalization error in Recurrent Neural Networks.

In addition, in nearly all convolutional neural networks at every layer we use different matrices. The question then arises why do we not do the same in recurrent neural networks. As mentioned in the Deep Learning Book(Goodfellow,Bengio,Courville) the idea behind using one Matrix is that at each time step we want to process information the same way. One could argue that we could rephrase the statement above as processing every two time steps the same way, or processing every three time steps the same way. This would allow for one to have multiple matrices as well. Thus we try the same thing

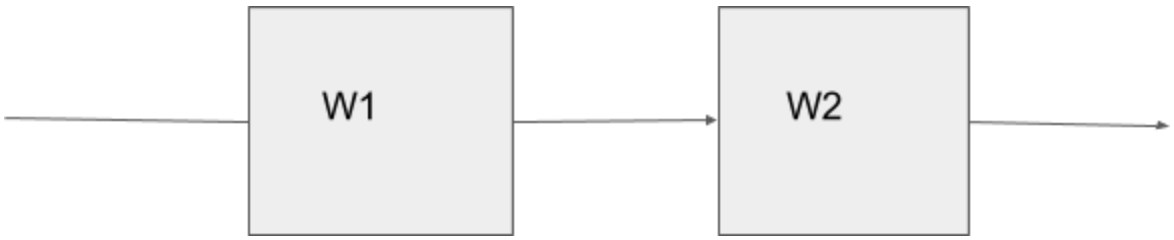
Approach

Below we abstract the type of models we are going to try in order to test the power of skip connection and multiple matrices. Here are the abstracted models we will try.

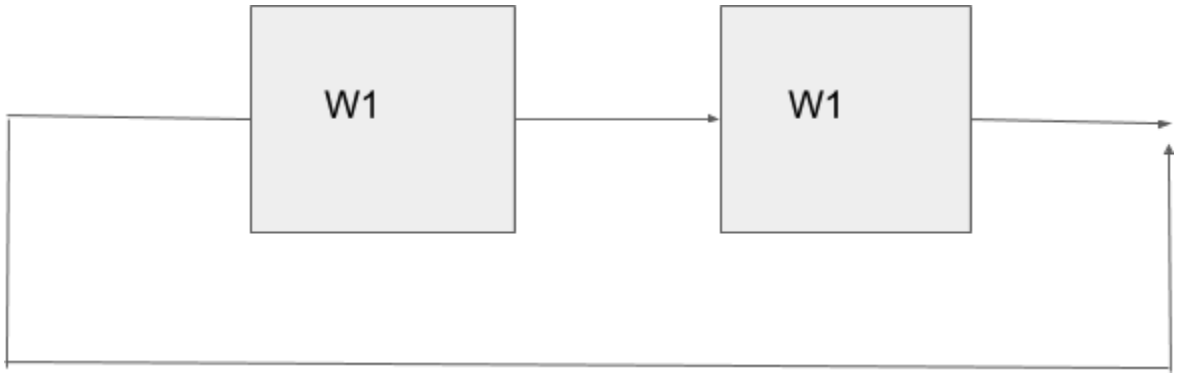
Version A: This is an ordinary LSTM or RNN. The W1 is here to demonstrate we only use one set of matrices. Thus at each time step h(t) = Relu(W1 * h(t-1) + Wx * Input).

Version B: A sequential RNN or LSTM network with alternating sets of matrices. Thus the h(t) equations alternate for different time steps. The equations alternate between h(t) = Relu(W1

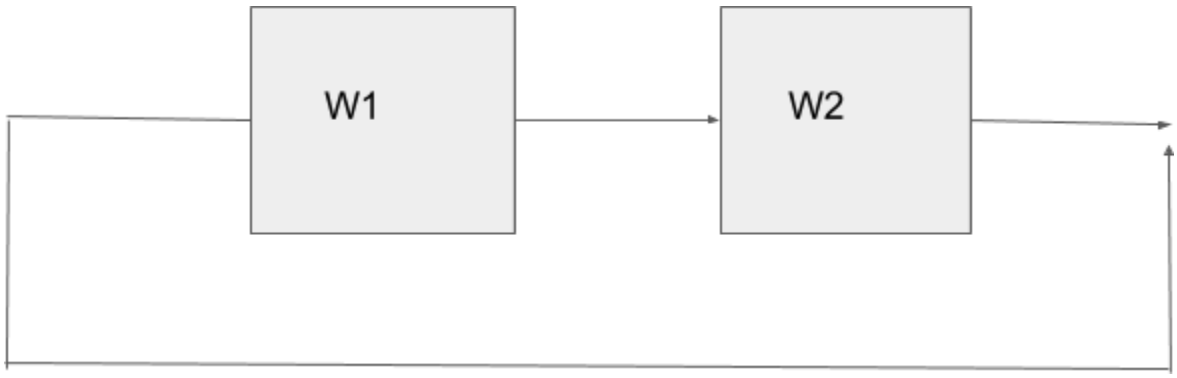
*h(t-1) + Wx * Input) and h(t) = Relu(W2 *h(t-1) + Wx * Input). Note that the matrix interacting with the input is not changing.



Version C: A sequential RNN or LSTM network with one set of matrices but skip connections skipping every two. Thus the equations alternate between the following. First h(t) = Relu(W1*h(t-1) + Wx*Input). Then h(t) = Relu(W1*h(t-1) + Wx*Input) + h(t-2)



Version D: Two sets of matrices with skip connections over every two blocks. Thus the equations alternate between the following. First h(t) = Relu(W1*h(t-1) + Wx*Input). Then h(t) = Relu(W2*h(t-1) + Wx*Input) + h(t-2)

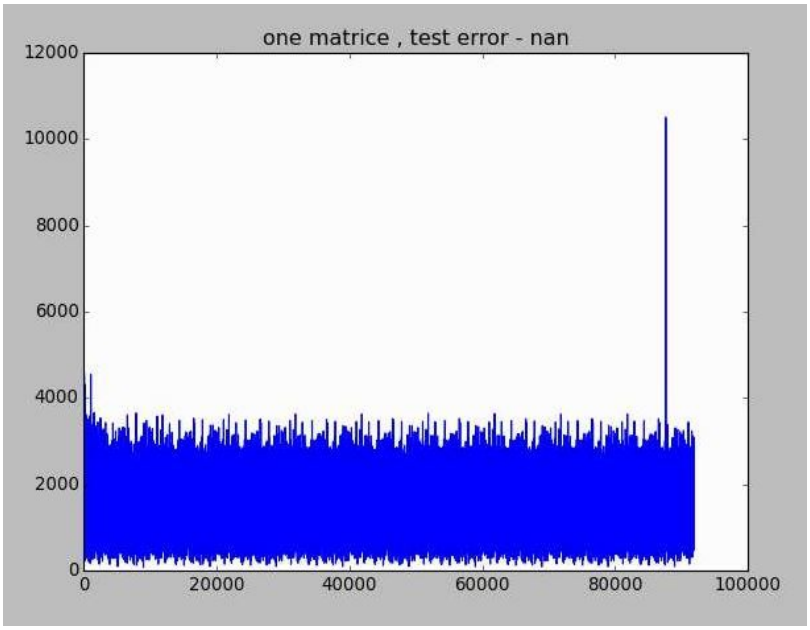


We will of course also have to regularize these while training!

Performance on Synthetic Dataset/Task

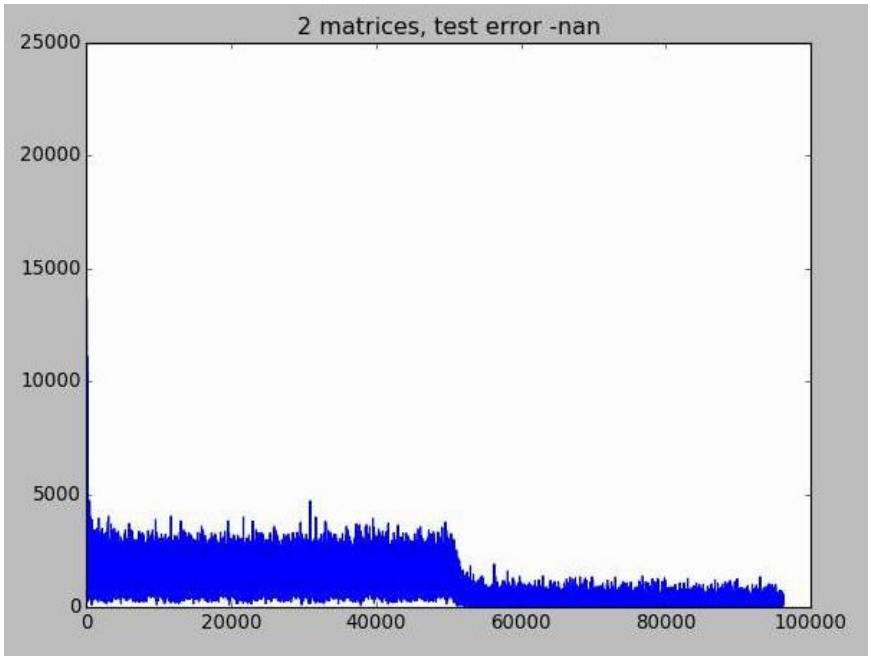
Version A: Here we show the performance of the simple RNN network. We are trying to train a a simple RNN to remember the first element of the first input. For this task I know there is a perfect solution for the 2 matrix case. Thus I know for sure the neural network has potential to go to loss equal to zero.

The loss is measured as mean squared error. Each data point is the loss measured for every batch of data. We cycle through 10000 inputs(length 100 , 6 dimensional vectors) about 10 times. Here we can see the simple RNN is unable to learn the task.

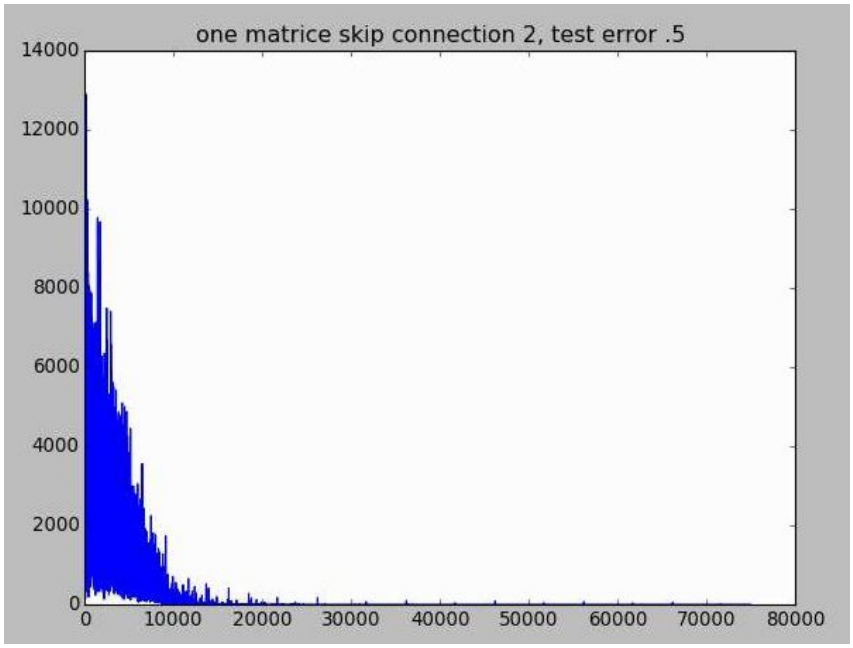


Version B:

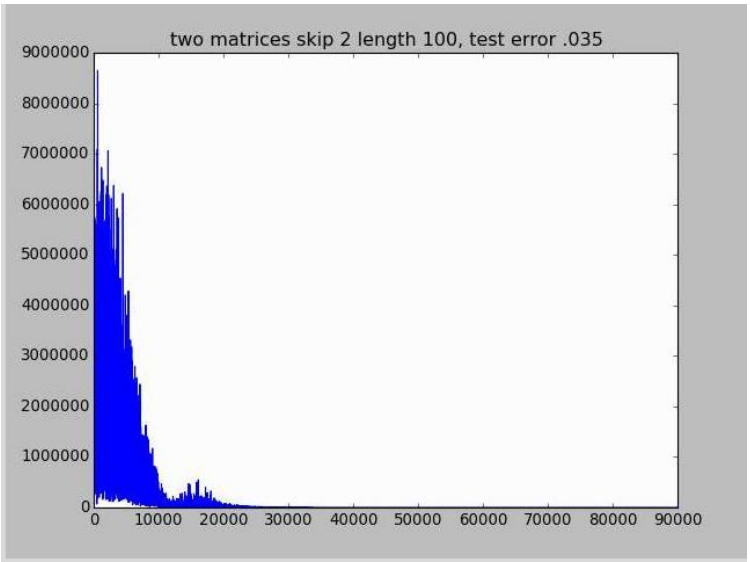
Same task and we see that once again a 2 matrix RNN can not learn anything either. In both version A and version B, the training eventually leads to computational issues and we get NaN for our test error. One interesting thing to note here there is a step in the graph drastically decreases to another level. One intuition I have for this behavior is that with 2 matrices there are more potential solutions that get closer to zero loss. In the 2 matrix case it is more likely as gradient descent ambles around it is more likely to set into a lower energy state!



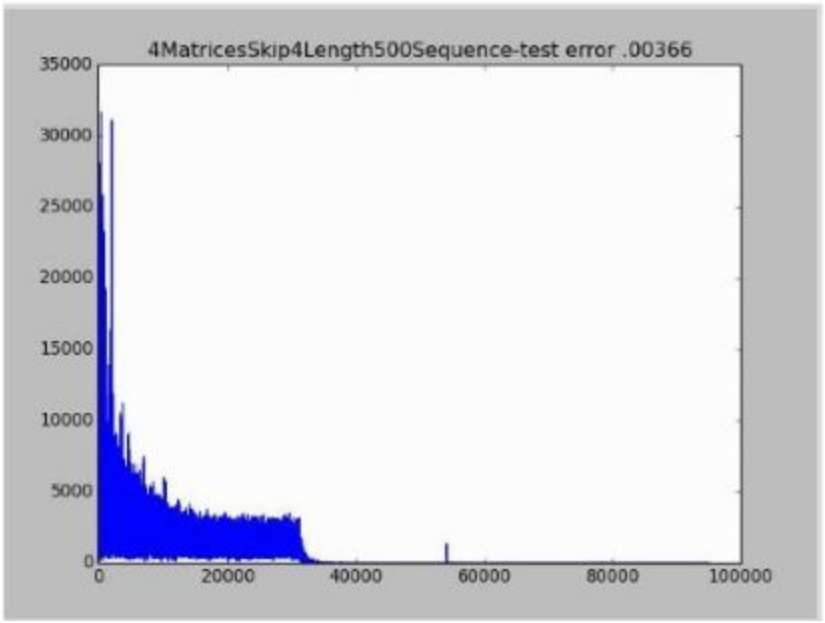
Version C: Same task but we see that the neural network with skip connections is doing very well in learning the task! Nearly after one epoch we are getting close to zero mean squared error. In addition there were no difficulties with this network giving NaN values. It is possible that skip connections have less exploding gradients!



Version D: Skip connections with two matrices and we can learn the task fast again! Here we also see the test error is a bit better than the one matrix case.

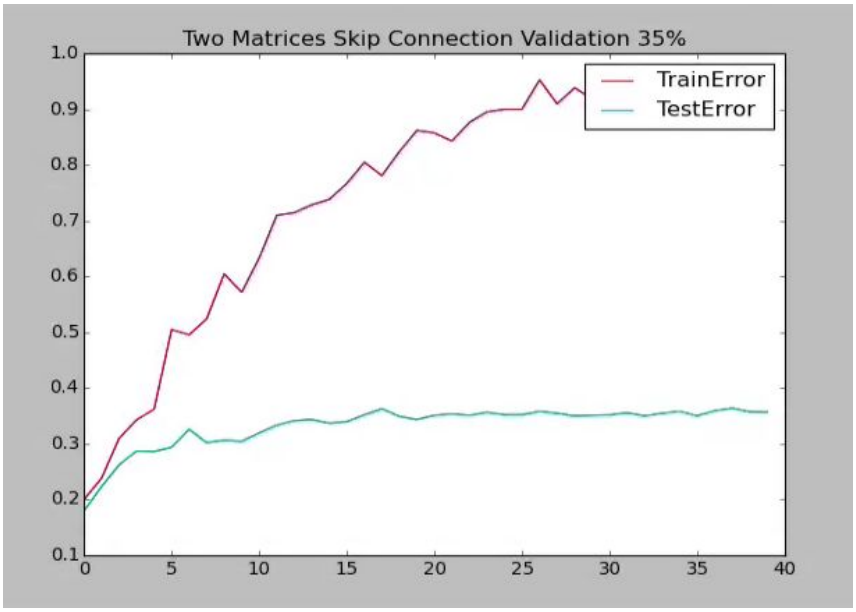


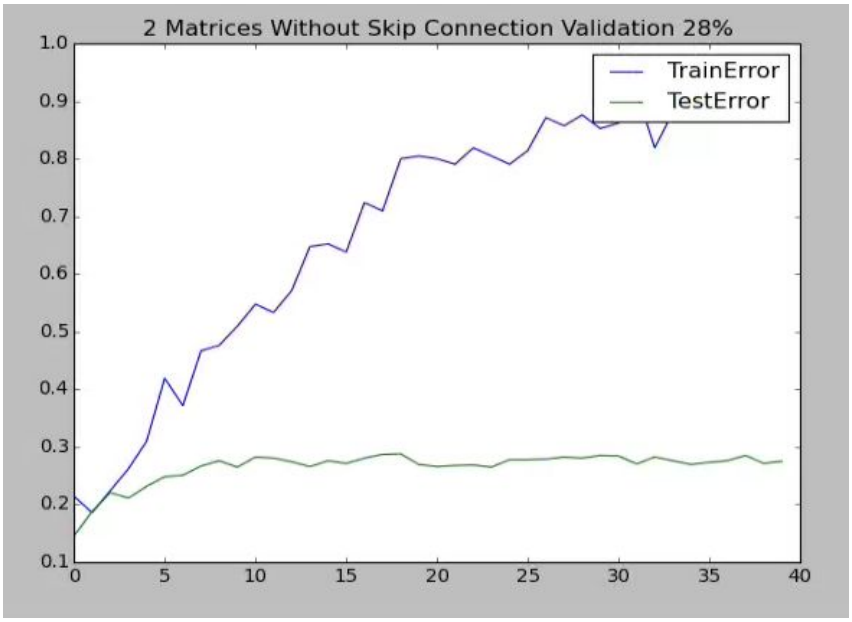
Special Version: Here Just for the sake of fun we tried train a recurrent neural network with 4 matrices and a skip connection that stretches every four time steps. We trained on an input sequence that is length 500 long! Here we are able to get to .00366 test error. This is amazing! Here once again I hypothesize it is able to do this because there are many 4 matrix solutions that get close to zero error.

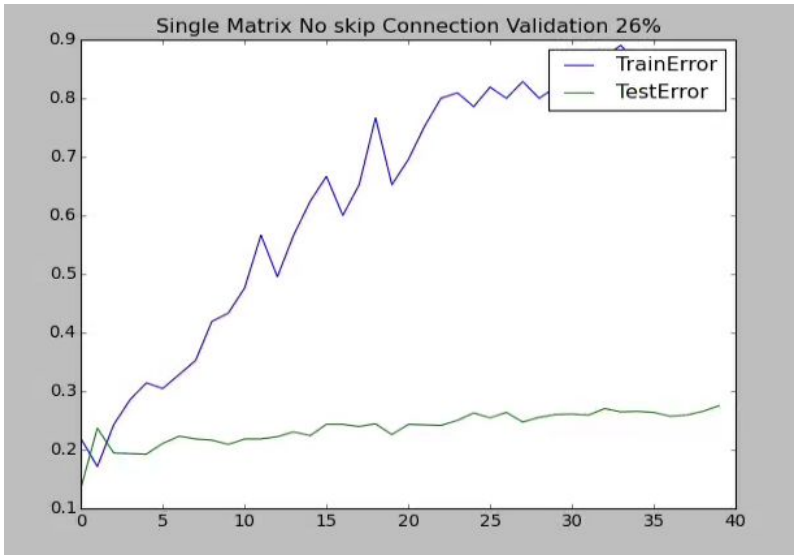


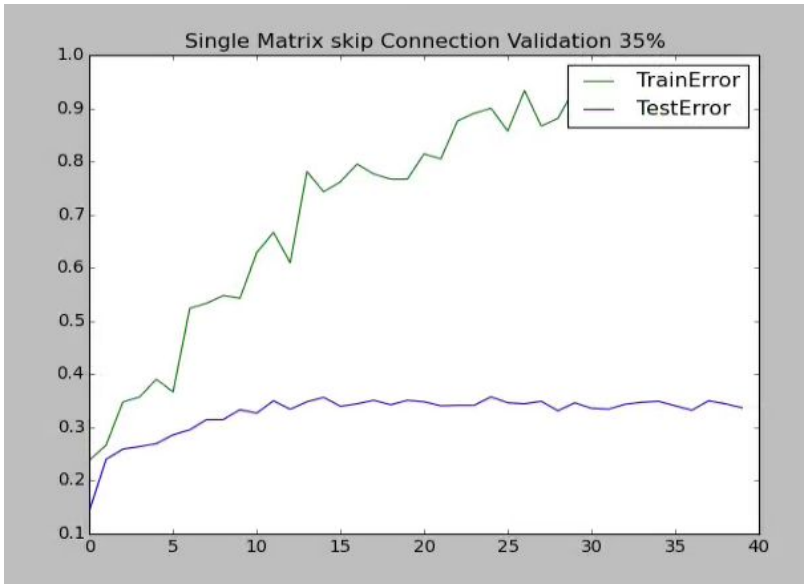
Thus overall skip connections are doing well on a synthetic task. Here we show a few graphs of results from the Stanford Sentiment Tree Bank Dataset one fine grained testing (i.e. 5 way classification).

Results for Basic RNNs



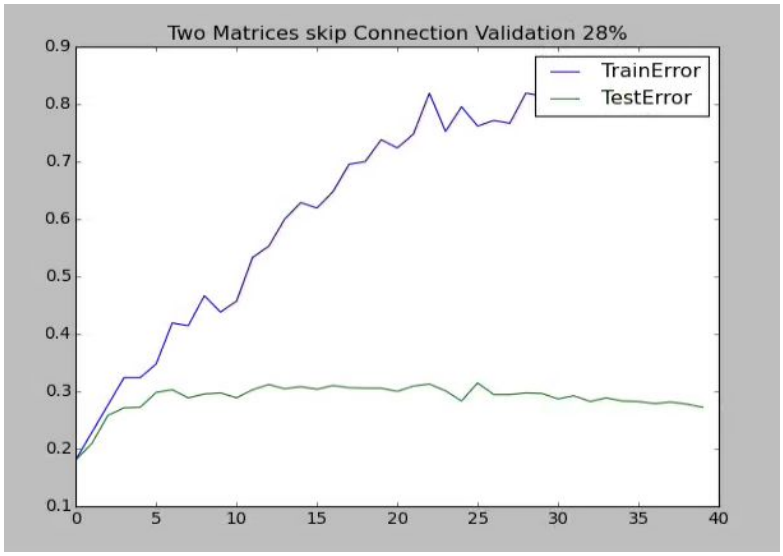


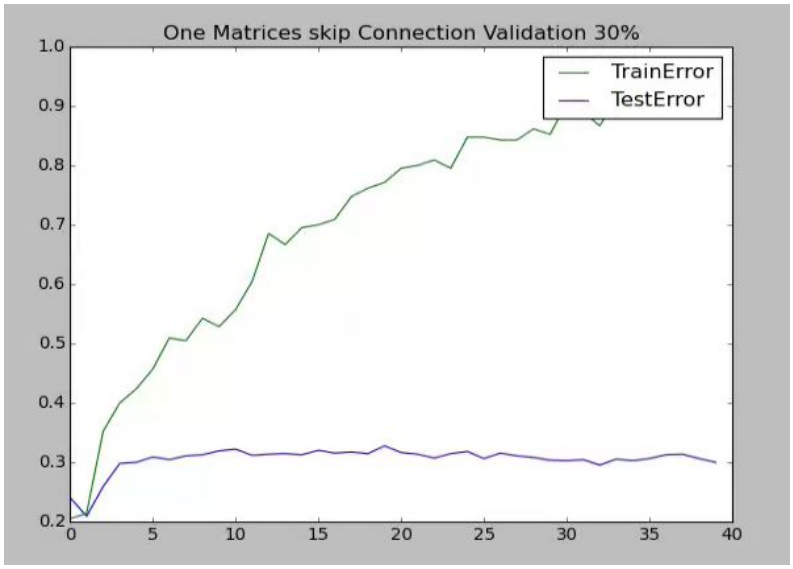


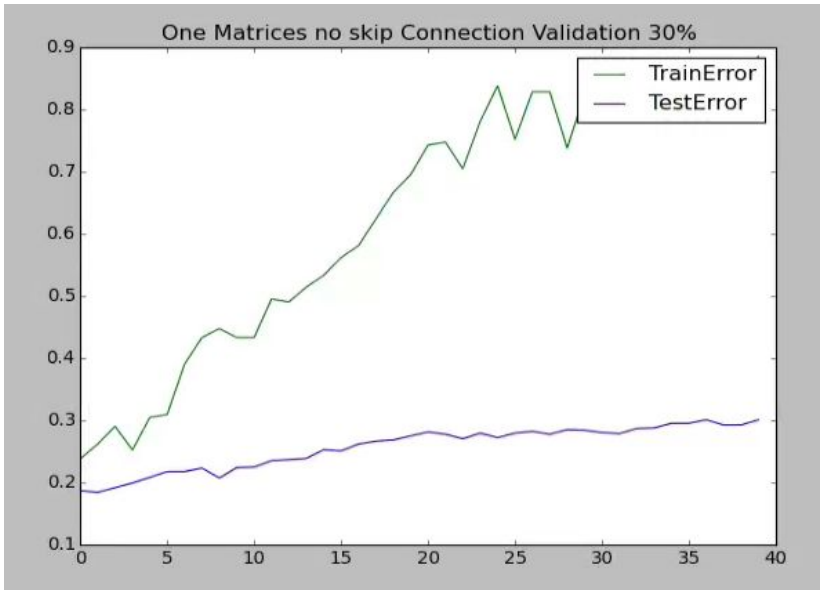


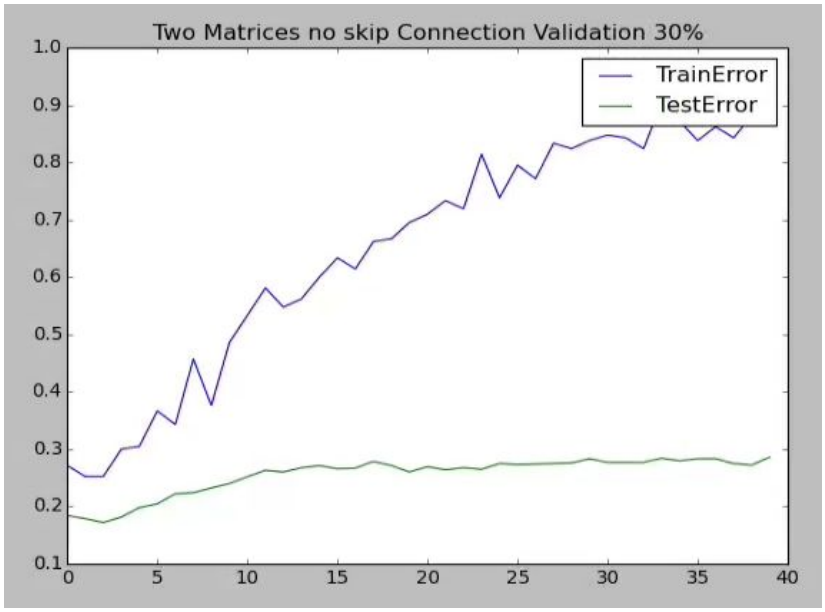
Overall we see here that skip connections for RNNs on the Stanford Sentiment Tree Bank are helping quite a bit. There is not however much difference between the two matrix case and the one matrix case. In addition regularization did not seem to help much.

Results for LSTMS


Two Matrices skip Connection Validation 28%
TrainError
TestError
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
5
10
15
20
25
30
35
40


One Matrices skip Connection Validation 30%
TrainError
TestError
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
5
10
15
20
25
30
35
40





In these results we can see that skip connections are not helping LSTMS peform better. What is interesting however is that RNNs paired with skip connections are performing better than LSTMs paired with skip connections. This however could be a bug in the code. LSTMS are usually able to get better performance on the Stanford Sentiment Tree Bank Dataset. I could not figure out why my LSTMs were not performing up to par.

Conclusion

There is no clear conclusion. It is clear that skip connections can be very useful in learning longer term tasks. It is clear that they even help on natural language processing problems. What is unclear however is how using multiple matrices is helping the problem. In some sense using one matrix is a good regularizer. It might be the case that using 2 Matrices creates some type of overfitting issues. My experiments also suggest that sometimes using RNNs with skip connections are equally as good or better than LSTMS. Since RNNs use less computation than LSTMs it can often be quite advantageous to just use RNNs if issues like memory and speed are a concern. This could definitely be the case in something like mobile computing.

References

1) Deep Residual Learning for Image Recognition, Kaiming Hi et al., 2015
2) Deep Learning Book, Ian Goodfellow, Yoshua Bengio, Aaron Courville
3) Lin, T., Horne, B. G., Tino, P., and Giles, C. L. (1996). Learning long-term dependencies is not as difficult with NARX recurrent neural networks. IEEE Transactions on Neural Networks, 7(6), 1329–1338
4) Lin, Y., Liu, Z., Sun, M., Liu, Y., and Zhu, X. (2015). Learning entity and relation embeddings for knowledge graph completion. In Proc. AAAI'15 .
5) Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank Richard Socher et al