Assignment 3

ES14BTECH11009 | Karthik Yadav

1.

The highest testing accuracy achieved for RNNs was 97.59 % with 400,000 iterations and 95.66 % with 300,000 iterations. With 100,000 iterations, the maximum I could get was 94.62 %.

Hidden layers being 256 for all the above cases.

Other parameters :

```
learningRate = 0.0025
batchSize = 256
displayStep = 10
```

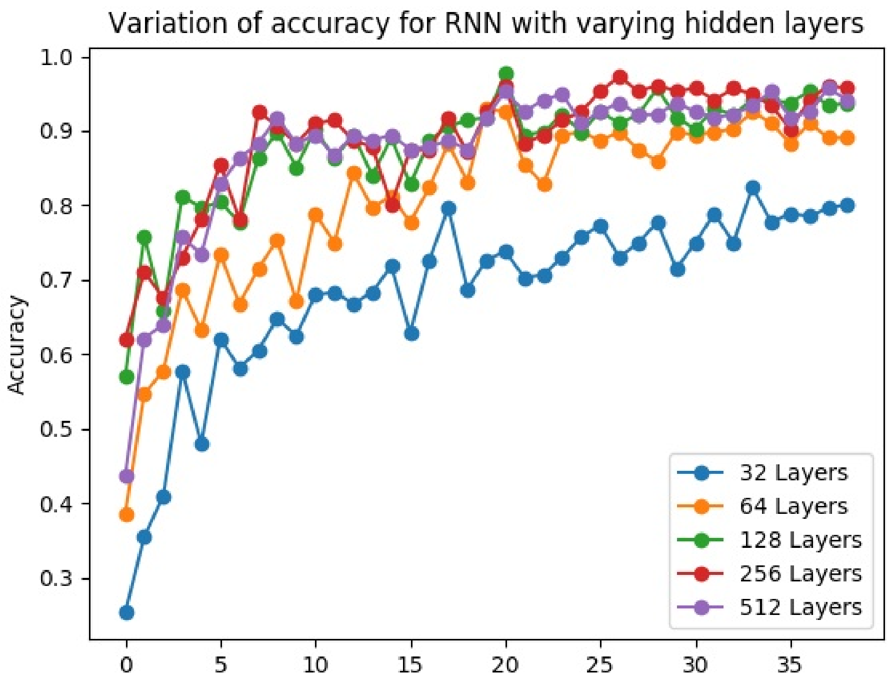
On testing with 512 hidden layers, the accuracy just dropped to 95.12 % and decreasing the hidden layers to 128 also brought down the accuracy to 96.39 %.

The cost function used was the recommended softmax cross entropy with logits, and the optimizer was an Adam Optimizer.

The learning rate was varied from 0.01 to 0.0025. Tests on 0.0001, gave much poor results.

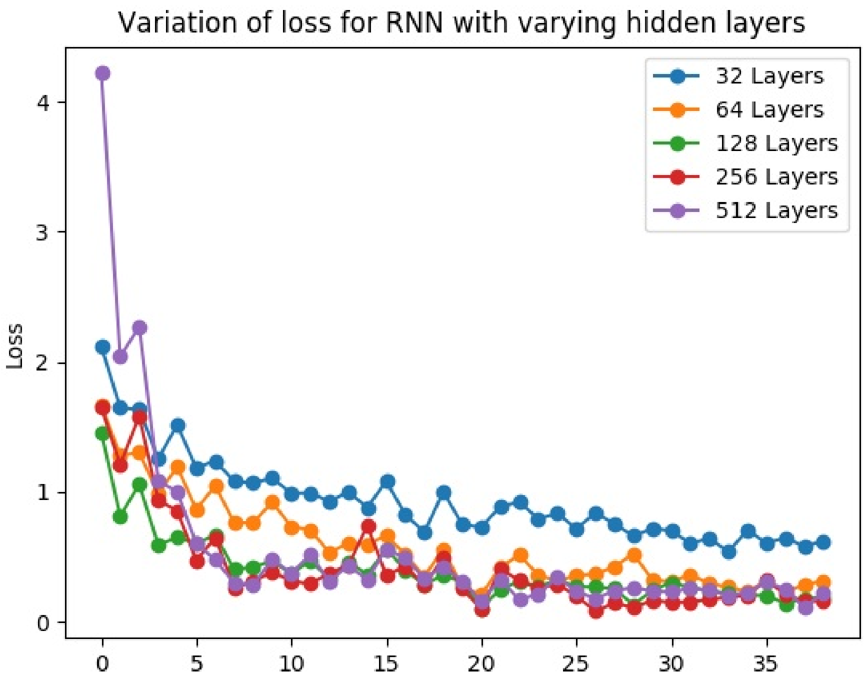
Batch size was played around between 128-256.

Hidden layers of lesser values gave lesser accuracies.



The above graph shows the variation of training accuracy for differnet layers for RNN.

We can clearly see from the above graph that 32 layers barely crossed 80 % training accuracy. 256 layers gives better accuracy than 512 layers.
But the highest value was achieved with 128 layers at 21st point.



Above graph shows the variation of training loss for differnet layers for RNN. 32 layers has the highest overall loss but 512 has the highest intial loss. We can see that the losses are still trying to stabilize, but there are some abrupt values in the middle.

2.

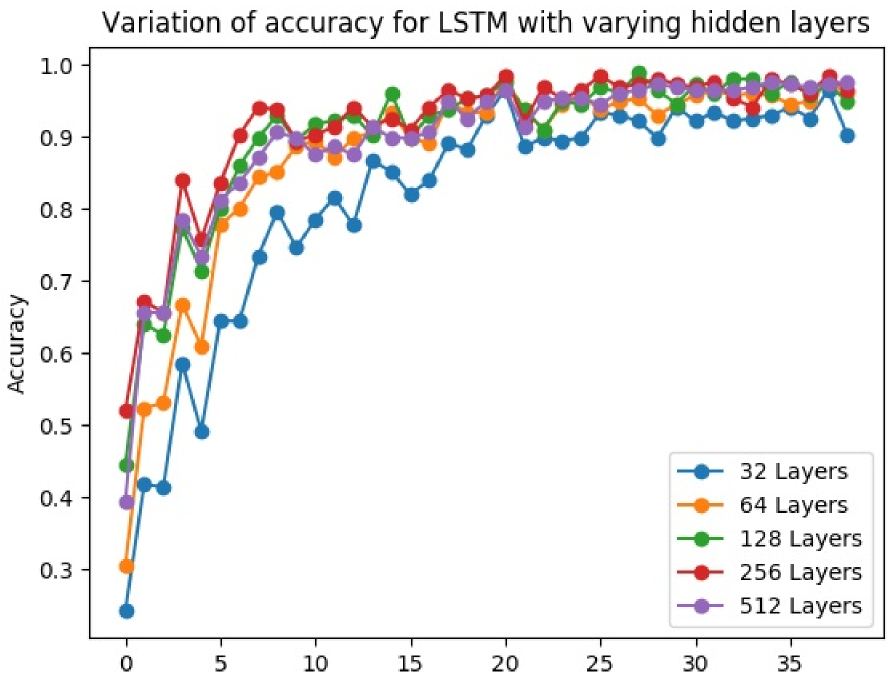
The test accuracy scores and also training accuracy/loss scores are much stable for LSTM and GRU compared to RNN.

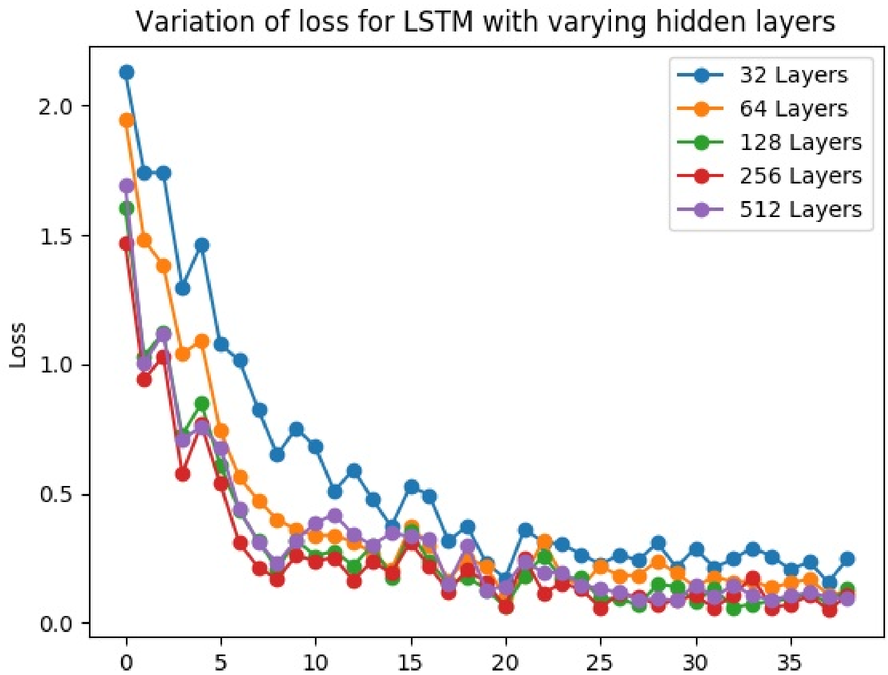
Hidden layers 128, 256 gave pretty good results. With 100,000 iterations,
LSTM gave a highest of 97.2 % with 256 layers and GRU gave a highest of 97.5 % with same layers and iterations.

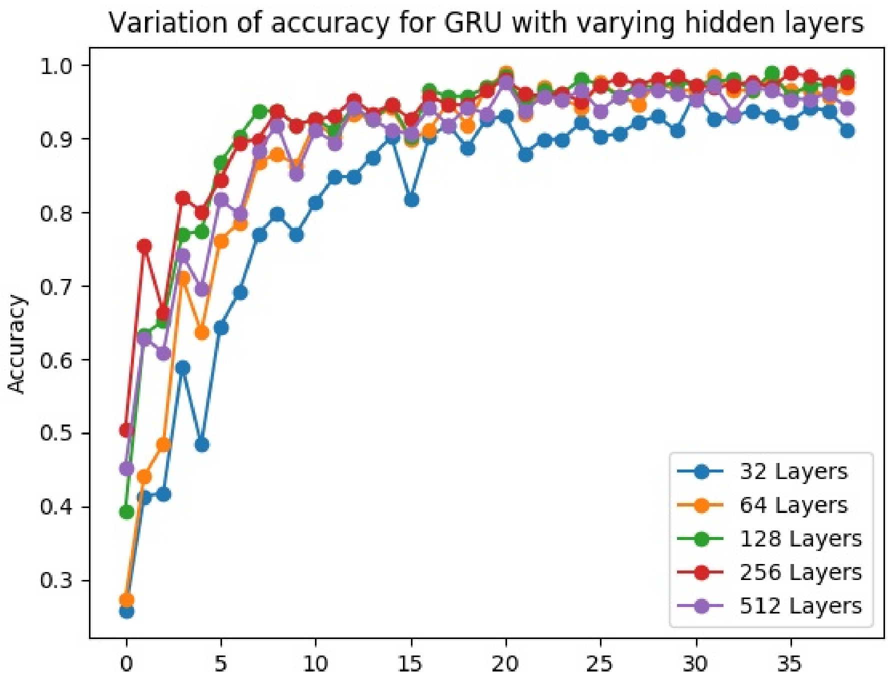
100,000 iterations were enough to get these higher values. Going to 200,000 iterations just overfits.

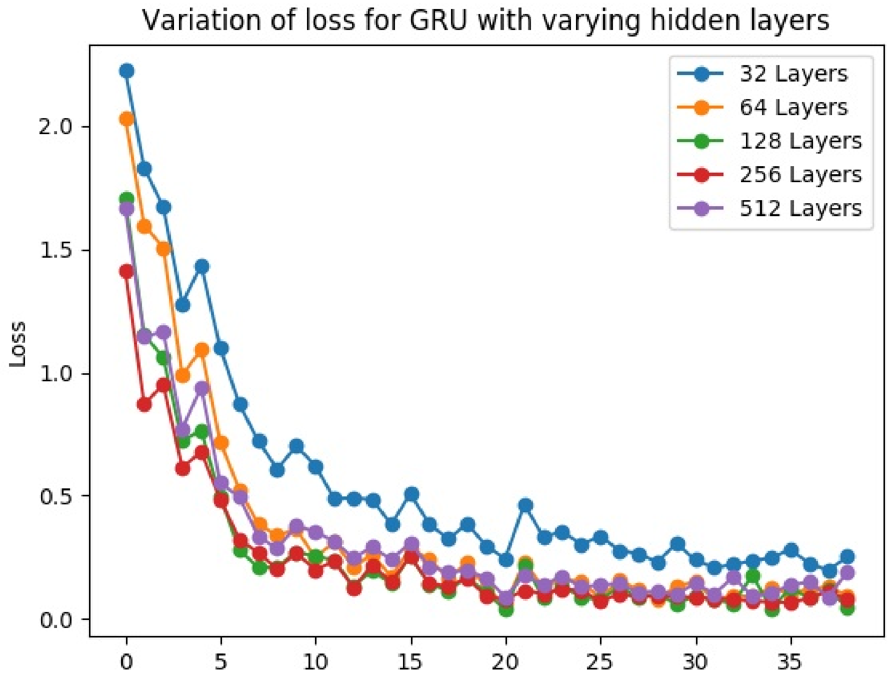
Other parameters :

learningRate = 0.0025
batchSize = 256
displayStep = 10


Variation of accuracy for LSTM with varying hidden layers
Accuracy
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
5
10
15
20
25
30
35
32 Layers
64 Layers
128 Layers
256 Layers
512 Layers


Variation of loss for LSTM with varying hidden layers
Loss
2.0
1.5
1.0
0.5
0.0
0
5
10
15
20
25
30
35
32 Layers
64 Layers
128 Layers
256 Layers
512 Layers

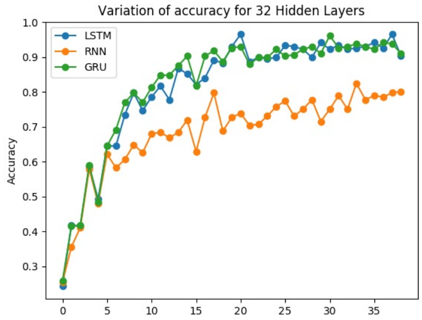


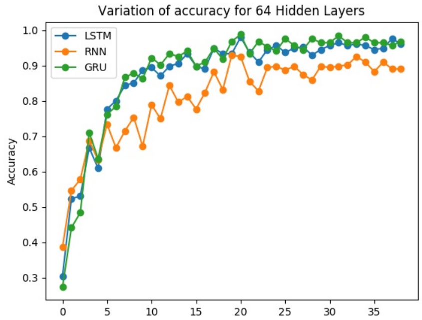


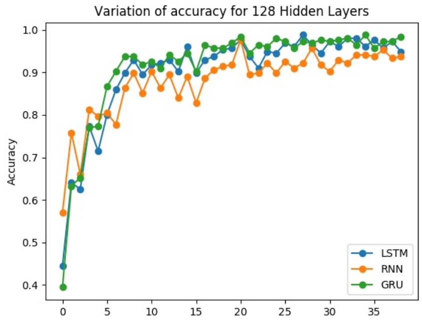
Above graph shows the variation of training accuracy/loss for differnet layers for LSTM and GRU. Almost all the layers give pretty stable results. Losses are also stable in the end. GRU has more stable compared results to LSTM.

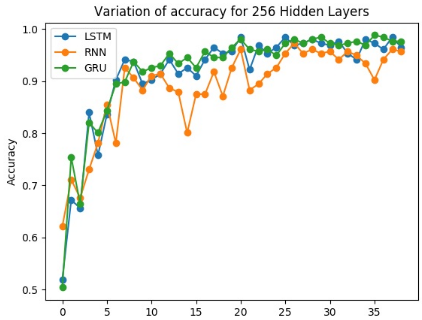
Only for 32 hidden layers, the values are a bit off compared to others.

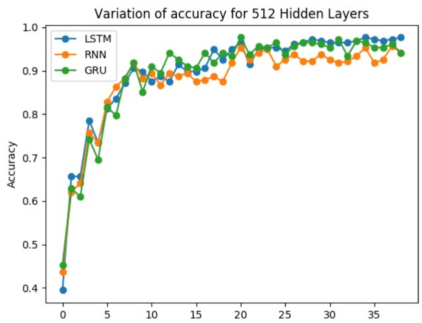
Below graphs are a comparion of training accuracy between LSTM/GRU/RNN for different layers.



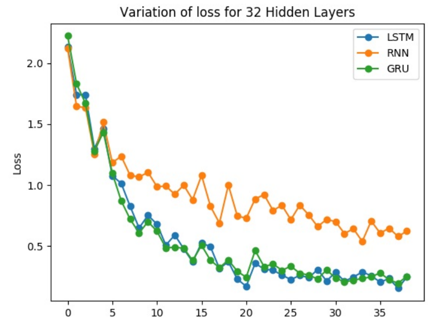


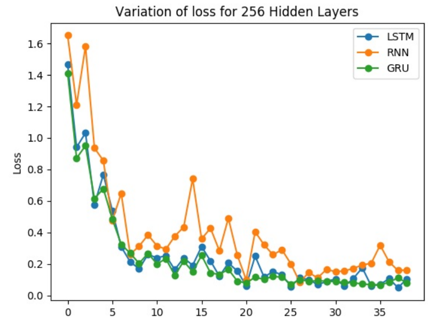






Below graphs are a comparion of training loss between LSTM/GRU/RNN for different layers.





3.

The RNN with a basic RNN cell trained much faster. LSTM and GRU took much longer time.

CNNs achieved the highest accuracy with few epochs.

For running CNNs, I used the default LeNet architecture CNNs for MNIST.
It has 2 convolution layers and 2 Fully Connected layers with 500 default batch size and Relu Activation function.

The CNN architectures converged in a much more stable manner than the RNN while holding the number of epochs constant.

The accuracy was around 97.5 % and was the highest. I am assuming the higher accuracy is due to fact that the hyperparameters has been tuned to be the best by the researchers while the hyperparameters for my RNN/LSTM/GRU were only experimented a few. Hence probably the difference.

Test accuracy for LSTM/GRU/RNN for 32/64/128/256/512 hidden layers :

```
test_acc_lstm_32 ('Testing Accuracy:', 0.93089998)
test_acc_rnn_32  ('Testing Accuracy:', 0.80129999)
test_acc_gru_32 ('Testing Accuracy:', 0.93529999)

test_acc_lstm_64  ('Testing Accuracy:', 0.95819998)
test_acc_rnn_64     ('Testing Accuracy:', 0.9059)
test_acc_gru_64 ('Testing Accuracy:', 0.96609998)

test_acc_lstm_128 ('Testing Accuracy:', 0.97070003)
test_acc_rnn_128  ('Testing Accuracy:', 0.94019997)
test_acc_gru_128    ('Testing Accuracy:', 0.96829998)

test_acc_lstm_256 ('Testing Accuracy:', 0.97210002)
test_acc_rnn_256  ('Testing Accuracy:', 0.94620001)
test_acc_gru_256    ('Testing Accuracy:', 0.97500002)

test_acc_lstm_512 ('Testing Accuracy:', 0.96860009)
test_acc_rnn_512  ('Testing Accuracy:', 0.93830019)
test_acc_gru_512    ('Testing Accuracy:', 0.96580017)
```