ELEC/COMP – 576 Assignment – 2 Report

NetID: sv38@rice.edu

Date: October 27, 2021

Task – 1: Visualizing a CNN with CIFAR10:

1-b: Train LeNet5 on CIFAR10:

Learning-Rates – 1e-3, 2e-2

Training Methods – AdamOptimiser, GradientDescentOptimiser

Training showed good results in the following hyperparameters:

Learning Rate: 1e-3

Training Method: AdamOptimiser

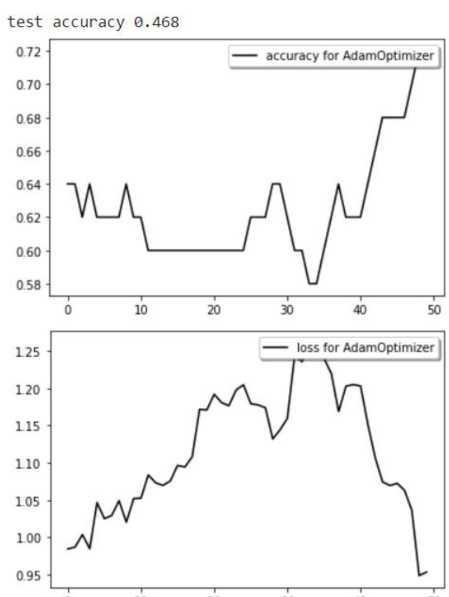


Fig – 1: AdamOptimiser, LR-1e-3

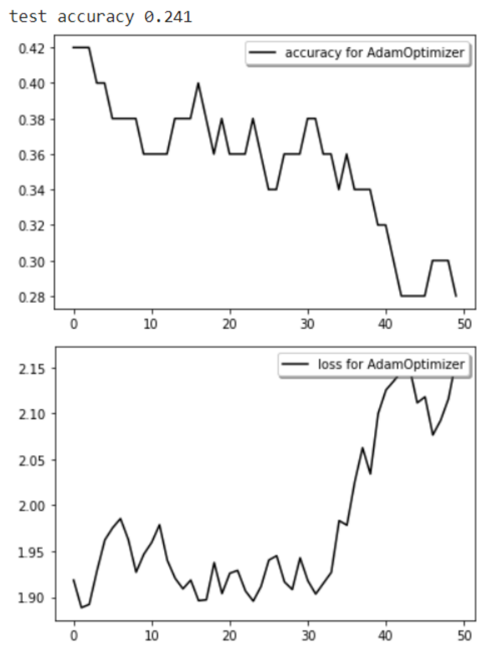
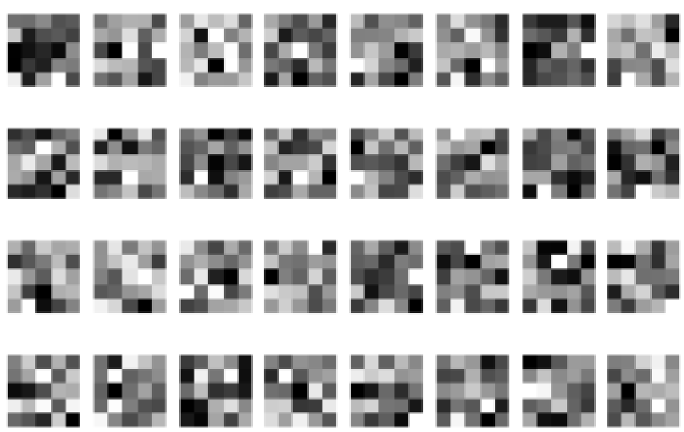


Fig-2: GradientDescentOptimiser, LR-1e-3

<u>1-c:</u> <u>Visualize the Trained Network:</u>



```
activation1: mean -0.00228988, variance 0.0049537
activation2: mean -0.0698792, variance 0.0113675
```

Fig-3: With AdamOptimiser

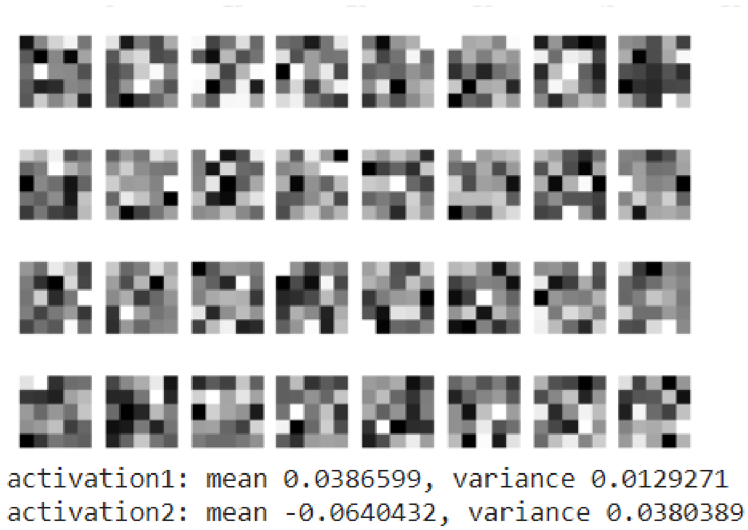


Fig-4: With GradientDescentOptimiser

Task – 2: Paper Summarization:

The paper "Visualizing and Understanding Convolutional Networks" addresses about the reason why Large Convolutional Networks perform good in ImageNet Benchmark. The authors describe about the reasons why the convolutional networks work and how to visualize them and how they achieve great performance.

So, they proposed a visualization method which not only works on the 1st layers but on the later layers also. The visualization the authors have made ore on the basis of mapping of Deconvnet, here they use the top-down approach to visualize their network and understand it.

Using the deconvnet they approach the feature activations on the Imagenet Validation set on a fully trained model. They have also deconstructed the Krizhevsky's network and selected suitable layers so that their method will be more effective for their approach on haw the convolution networks work.

Finally, they trained a model in the ImageNet 2012 and showed the resulted images of the visualization set where it showed the test results. These results help us in the featuring that networks do not have random and unpredictable patterns but are based valuable data collected in every layer.

Task – 3: Build and Train an RNN on MNIST:

3-a: Set Up a RNN:

Please see the code for the values of the parameters.

3-b: How about using an LSTM or GRU:

Here we ran 3 basic MNIST models:

- Basic LSTM Cell

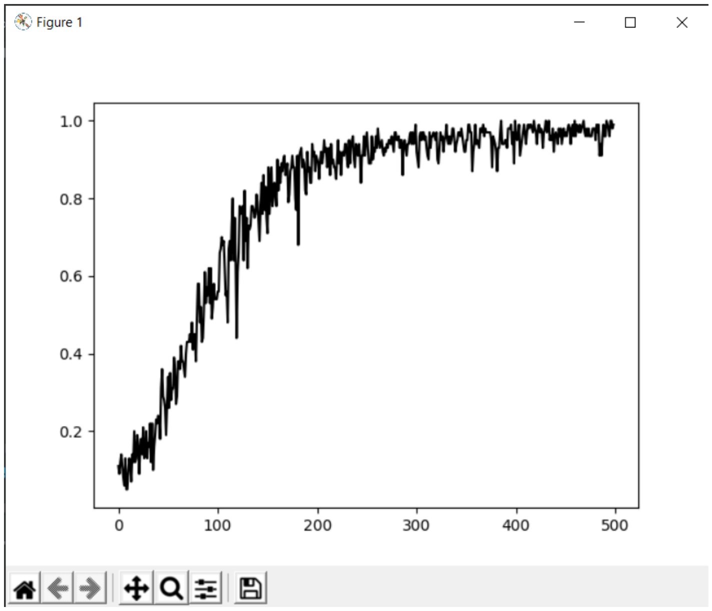


Fig-5 LMST cell Accuracy

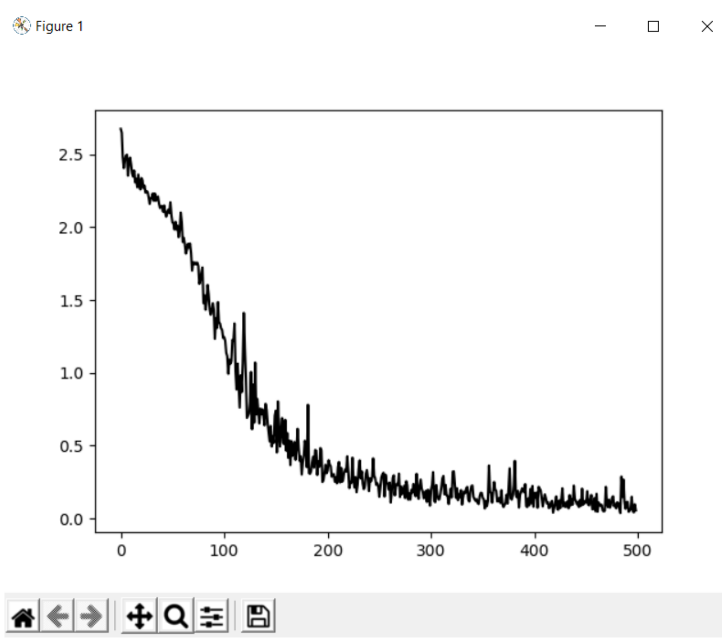


Fig-6 LMST Loss

```
Testing Accuracy: 0.9502
```

- GRU Cell

Testing Accuracy: 0.9373

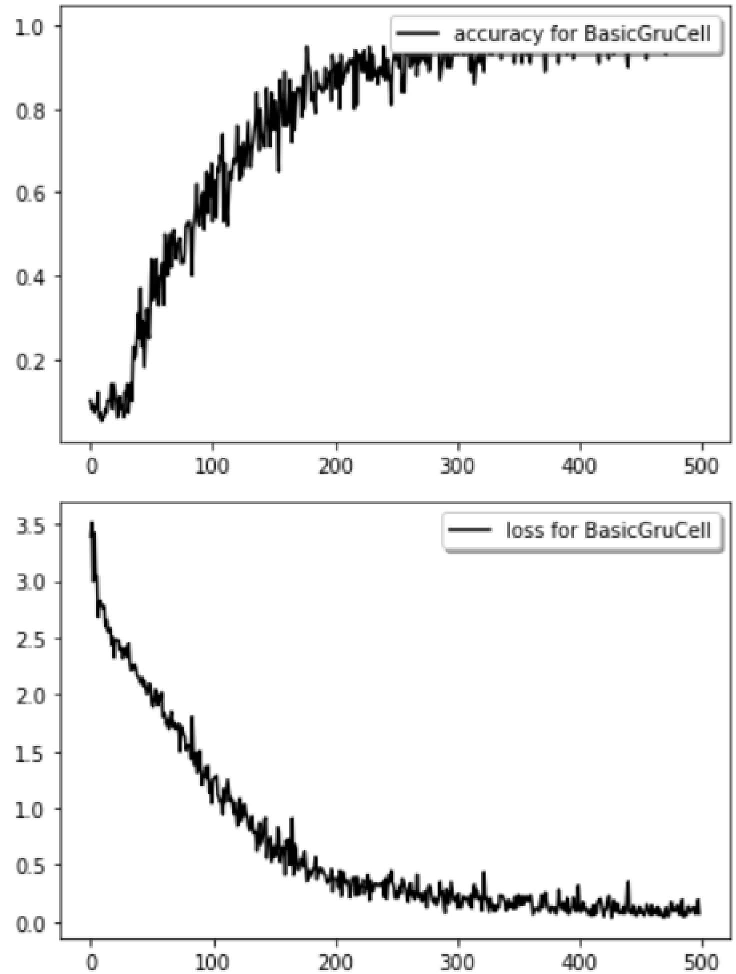


Fig-7 Basic GRU Cell

- Basic RNN cell

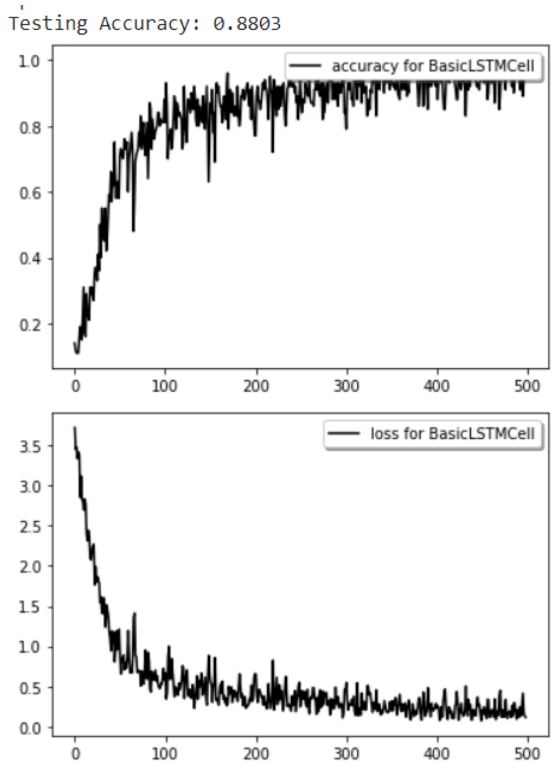


Fig-8 Basic RNN Cell

As we can see the RNN cell performance is a little poor than the other cells.

The RNN under different No. of neuron:

- 64 Units

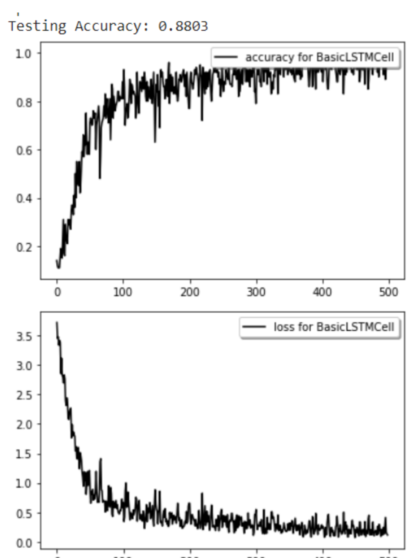


Fig-9 64 units

- 100 Units

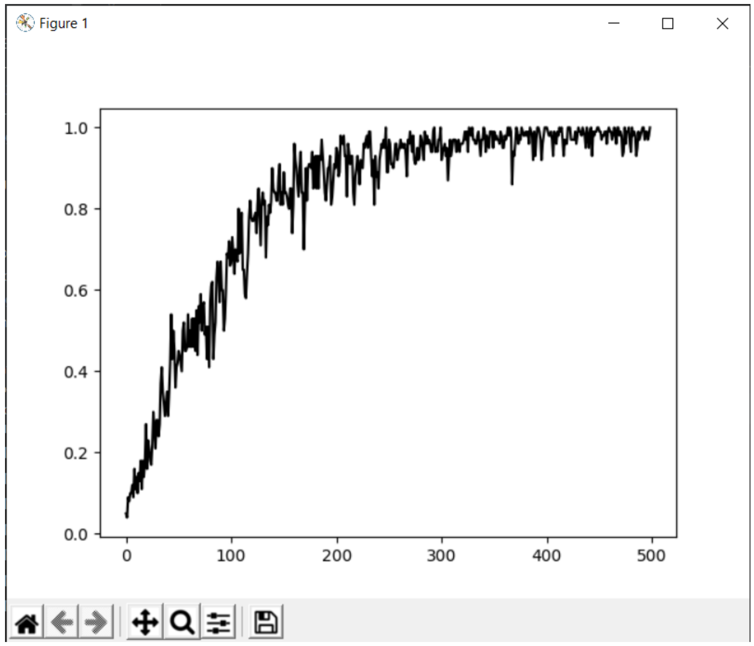


Fig-10 Accuracy for 100 units

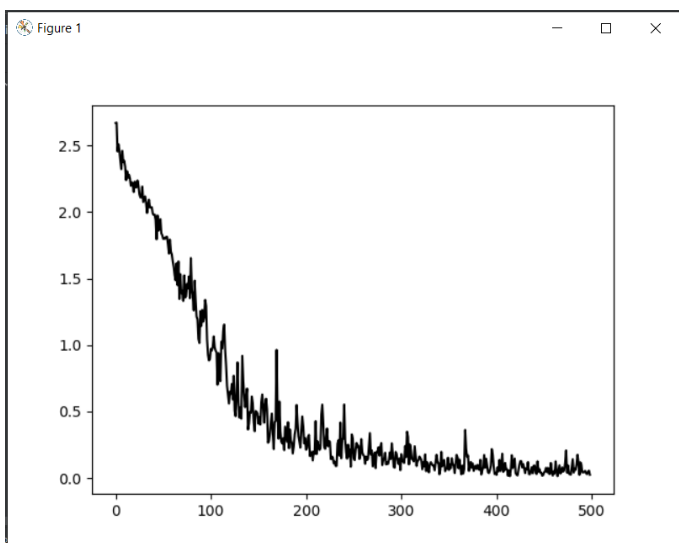


Fig-11 Loss for 100 units

- 256 Units

Testing Accuracy: 0.8981

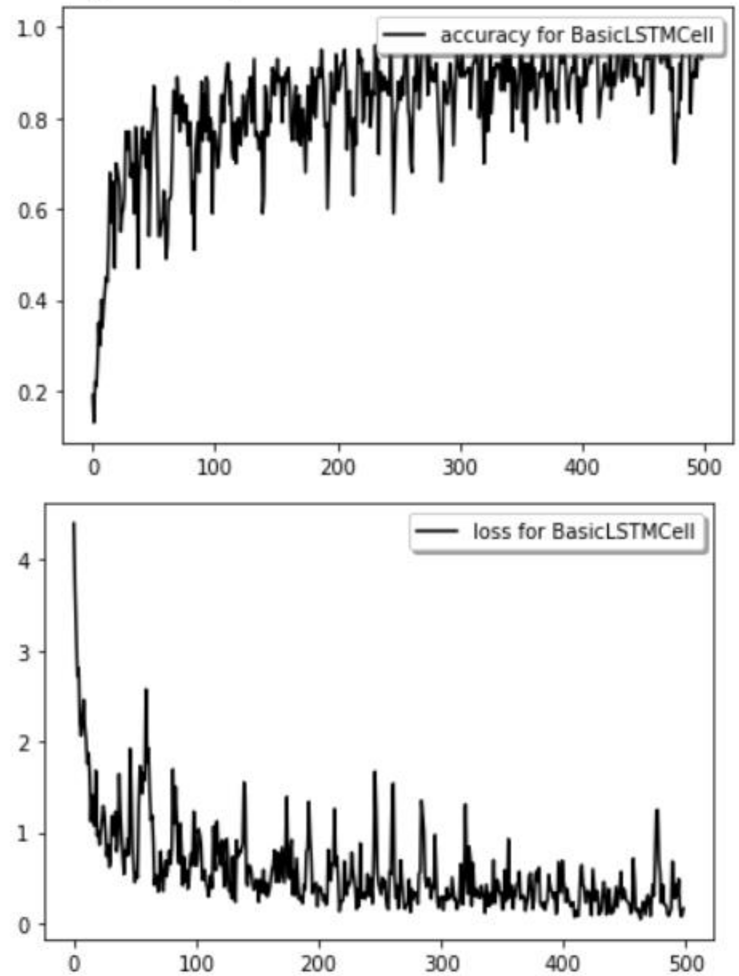


Fig-12 256 units

The cell runs better when the no. of neurons are 100.

3-c: Compare Against CNN:

Reference from - https://www.tutorialspoint.com/tensorflow/tensorflow_cnn_and_rnn_difference.htm

CNN:

1. It is suitable for spatial data.
2. It performs better than RNN
3. Inputs are fixed and generates fixed output sizes.
4. Ideal for images and video processing

RNN:

1. It is suitable for temporal data.
2. Has less features than CNN.
3. The inputs and outputs can be of any length.
4. I deal for text and speech analysis.