Report :: TIPR Assignment - II

Prasanna Patil

4 March, 2018

1 Configuration

- Python code is written for Python 3. It may not work with Python 2.
- The Cat-Dog images were rescaled to (28, 28) resolution and model was trained on grayscale images.
- The code should be executed from within src folder, otherwise relative paths may not work as expected.
- Two datasets are recognized as "MNIST" and "Cat-Dog" (without quotes).
- Pass the configuration as a string list. That is, argument should be –configuration '[30 10]' (**with** quotes).

2 Part 1:- MNIST

2.1 Task 1:- Test Model with different number of layers

Tested model with single, double and triple layers. Each having 30 number of neurons. Following plots show the various metrics for different layers.

Each plot shows the metric on Y-axis and epochs on X-axis. Different lines in plot corresponds to different models.

2.2 Task 2:- Test Model with different number of neurons and layers

Tested model with following configurations:

- 784-50-10
- 785-50-30-10
- 784-100-10
- 784-50-50-30-10

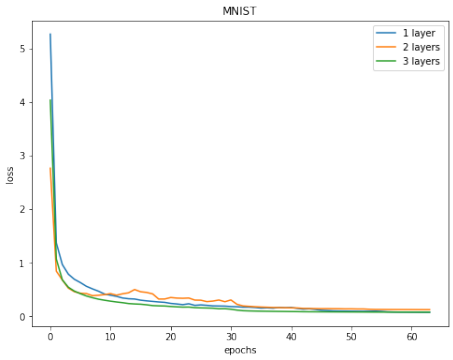

Figure 1: Loss

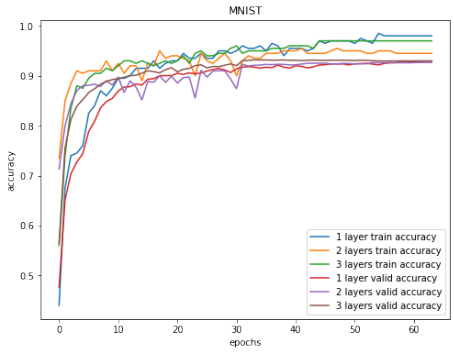

Figure 2: Accuracy

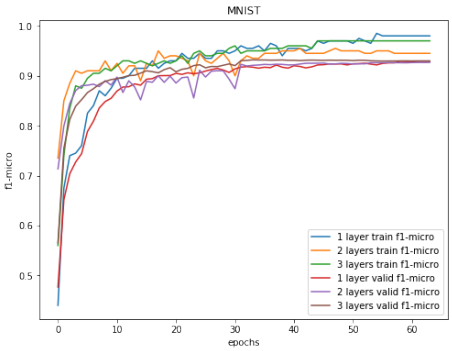

Figure 3: F1-Micro

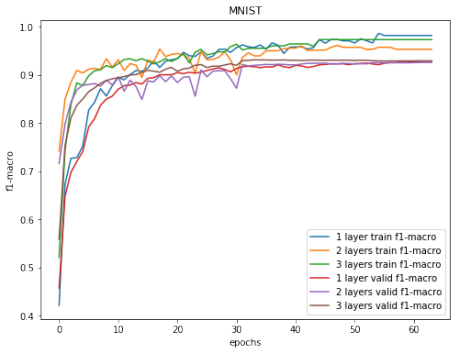

Figure 4: F1-Macro

Finally fixed and fine-tuned the 784-50-30-10 model as it took less time for training and had comparable performance.

Each plot shows the metric on Y-axis and epochs on X-axis. Different lines in plot corresponds to different models.

2.3 Task 3:- Test Model with different activation functions

Tested model with relu, tanh, sigmoid and swish activations functions. The performance of relu and swish were almost same and decided to use swish in the final model. For grayscale images tanh performed better than sigmoid in convergence rate.

Each plot shows the metric on Y-axis and epochs on X-axis. Different lines in plot corresponds to different models.

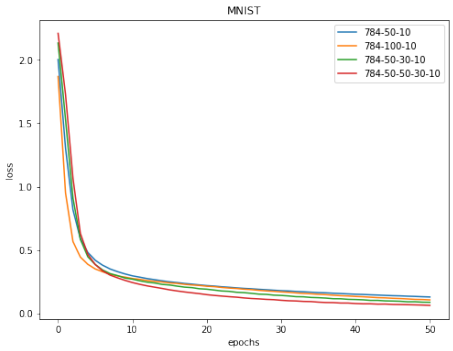

Figure 5: Loss

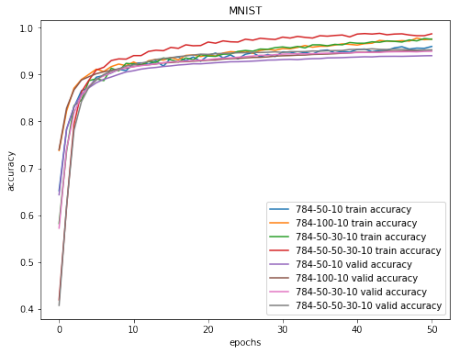

Figure 6: Accuracy

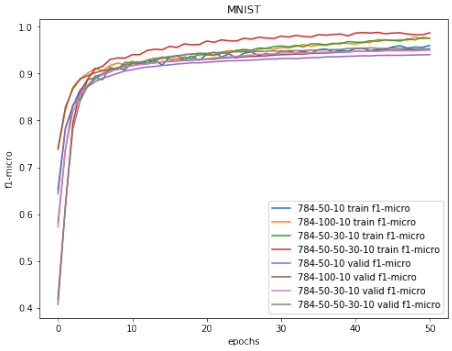

Figure 7: F1-Micro

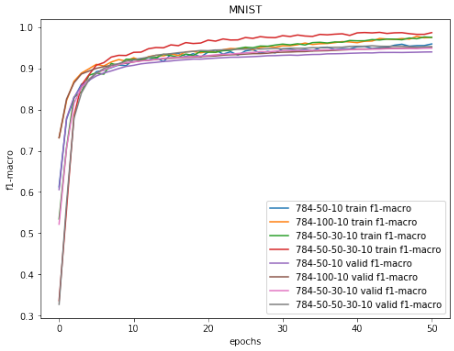

Figure 8: F1-Macro

2.4 Task 4:- Run Model with Keras code

Tested the same models as in above tasks but this time used keras to implement the Feed Forward Network.

2.4.1 Testing with different Layers

2.4.2 Testing with different architectures

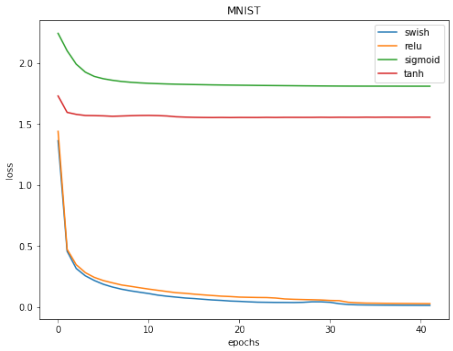

Figure 9: Loss

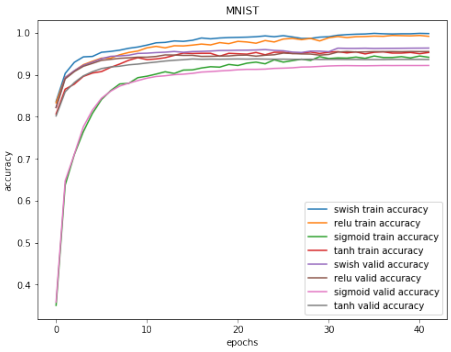

Figure 10: Accuracy

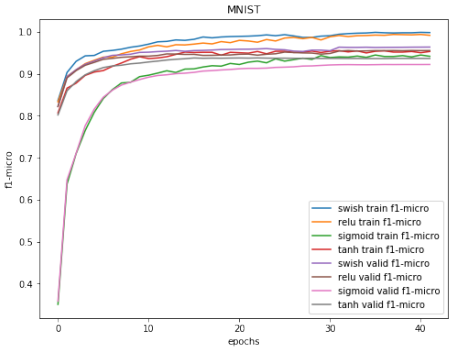

Figure 11: F1-Micro

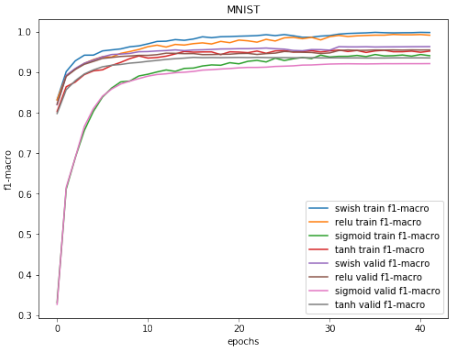

Figure 12: F1-Macro

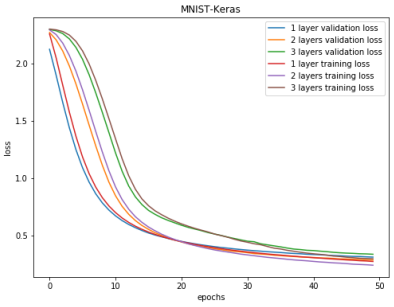

Figure 13: Loss

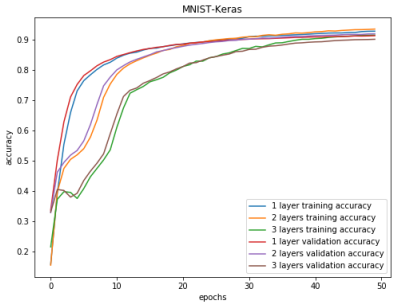

Figure 14: Accuracy

2.4.3 Testing with different activation functions

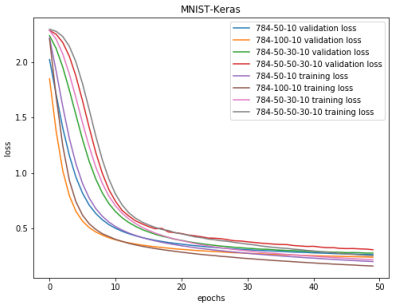

Figure 15: Loss

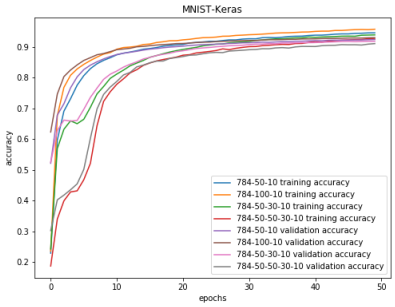

Figure 16: Accuracy

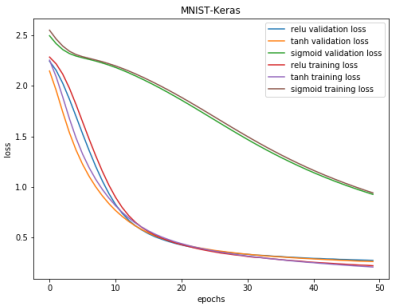

Figure 17: Loss

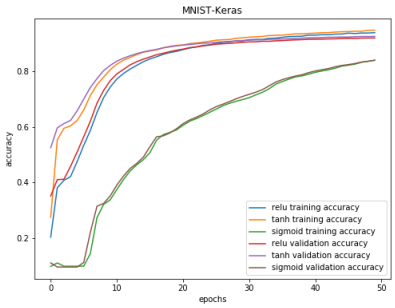

Figure 18: Accuracy

3 Part 2:- Cat-Dog

3.1 Task 1:- Test Model with different number of layers

Tested model with single, double and triple layers. Each having 200 number of neurons. Following plots show the various metrics for different layers.

Each plot shows the metric on Y-axis and epochs on X-axis. Different lines in plot corresponds to different models.

3.2 Task 2:- Test Model with different number of neurons and layers

Tested model with following configurations:

- 784-50-10
- 785-50-30-10
- 784-100-10

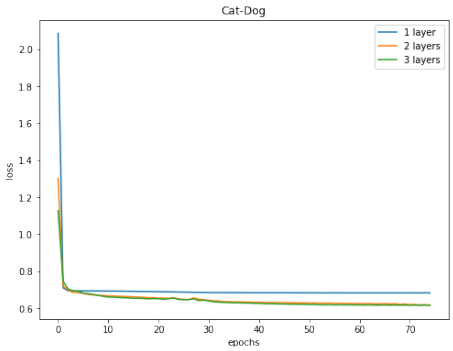

Figure 19: Loss

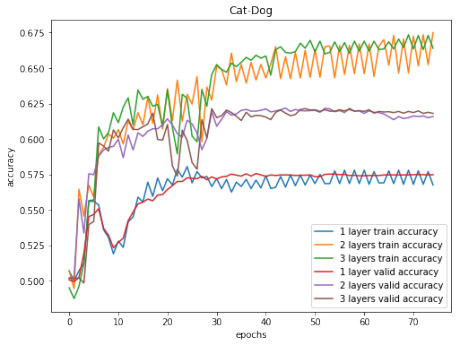

Figure 20: Accuracy

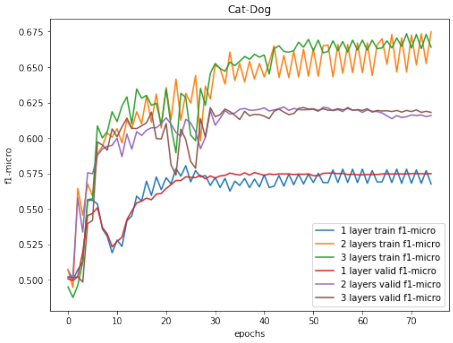

Figure 21: F1-Micro

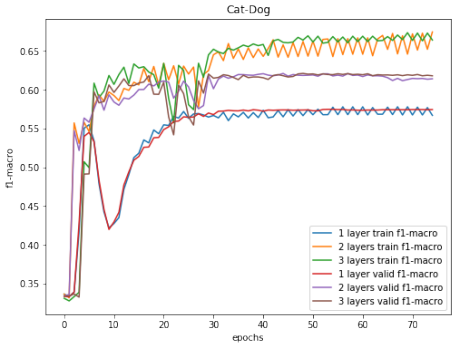

Figure 22: F1-Macro

- 784-50-50-30-10

Finally fixed and fine-tuned the 784-50-30-10 model as it took less time for training and had comparable performance.

Each plot shows the metric on Y-axis and epochs on X-axis. Different lines in plot corresponds to different models.

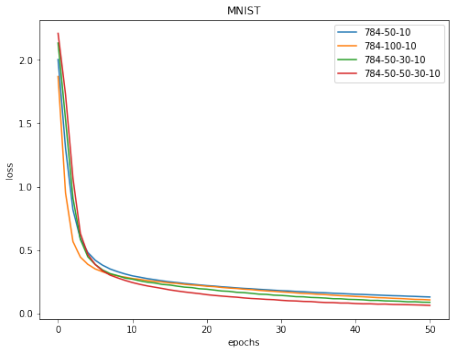

Figure 23: Loss

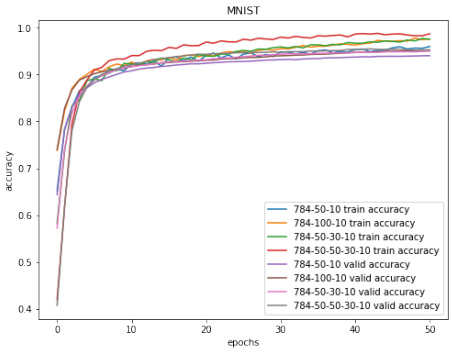

Figure 24: Accuracy

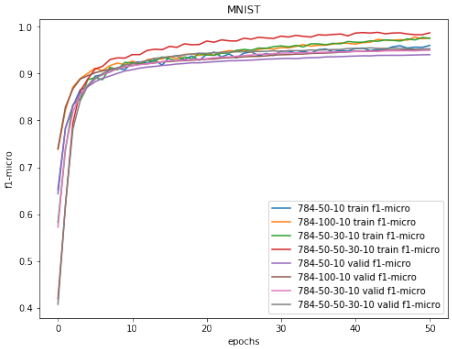

Figure 25: F1-Micro

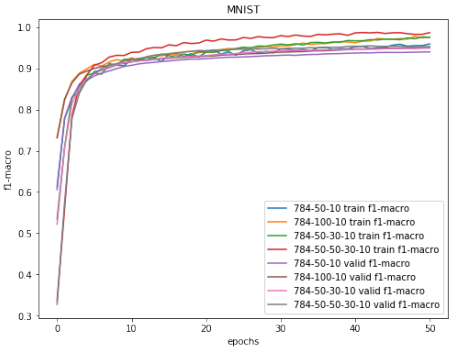

Figure 26: F1-Macro

3.3 Task 3:- Test Model with different activation functions

Tested model with relu, tanh, sigmoid and swish activations functions. The performance of relu and swish were almost same and decided to use swish in the final model. For grayscale images tanh performed better than sigmoid in convergence rate.

Each plot shows the metric on Y-axis and epochs on X-axis. Different lines in plot corresponds to different models.

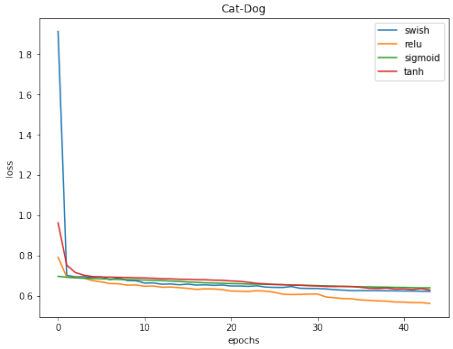

Figure 27: Loss

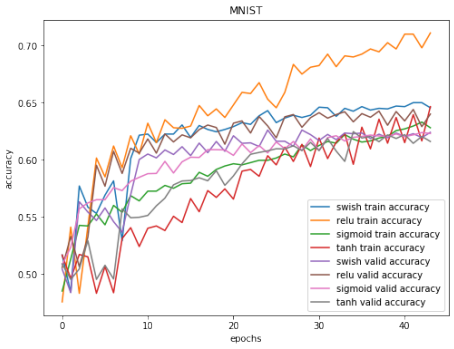

Figure 28: Accuracy

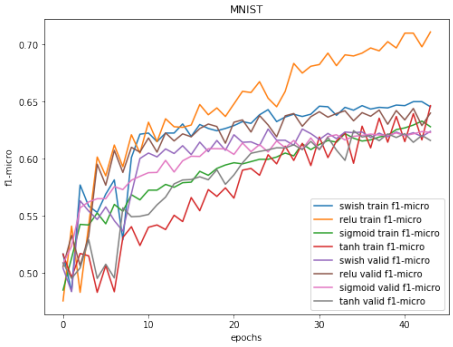

Figure 29: F1-Micro

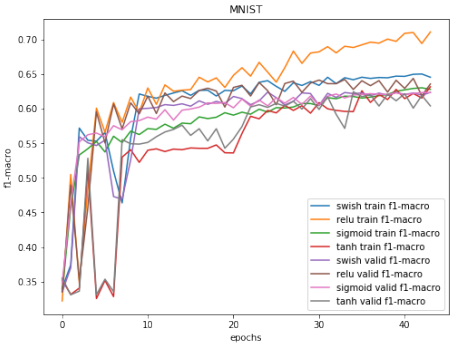

Figure 30: F1-Macro

3.4 Task 4:- Run Model with Keras code

Tested the same models as in above tasks but this time used keras to implement the Feed Forward Network.

3.4.1 Testing with different Layers

3.4.2 Testing with different architectures

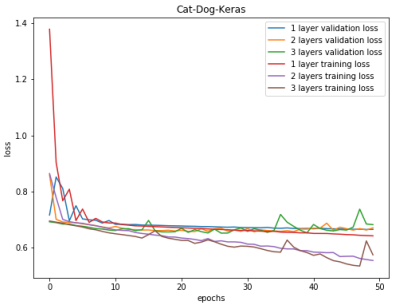

Figure 31: Loss

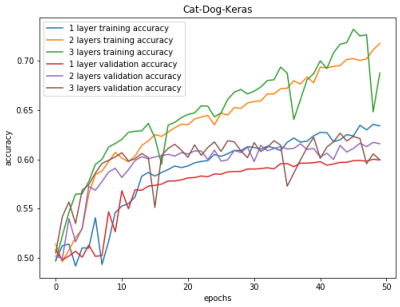

Figure 32: Accuracy

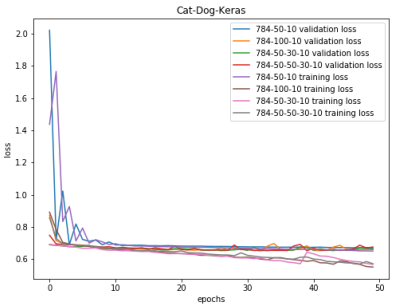

Figure 33: Loss

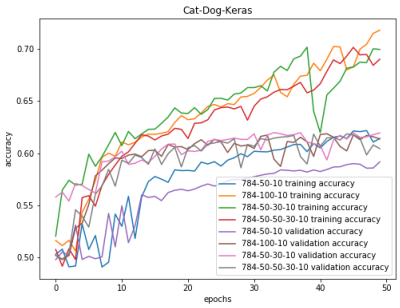

Figure 34: Accuracy

3.4.3 Testing with different activation functions

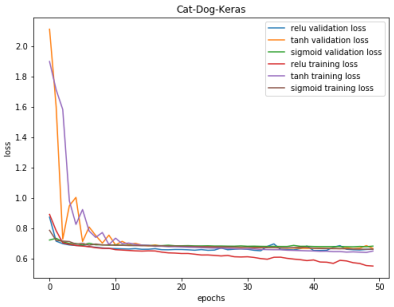

Figure 35: Loss

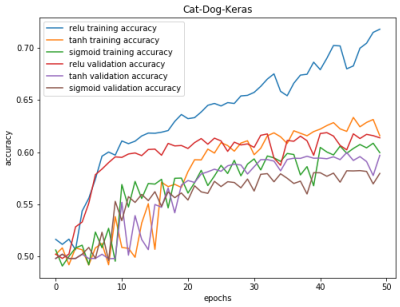

Figure 36: Accuracy

4 Part 3:- Pubmed, Twitter and Dolphins

4.1 PubMed Dataset

Following plot shows accuracy and loss when neural network was trained on pubmed dataset.

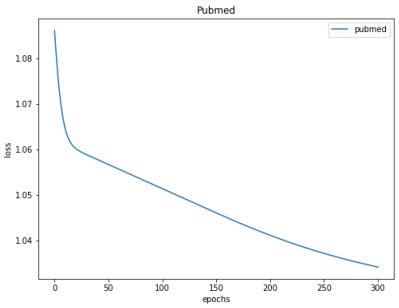

Figure 37: Loss

Figure 38: Accuracy

4.2 Twitter Dataset

Following plot shows accuracy and loss when neural network was trained on twitter dataset.

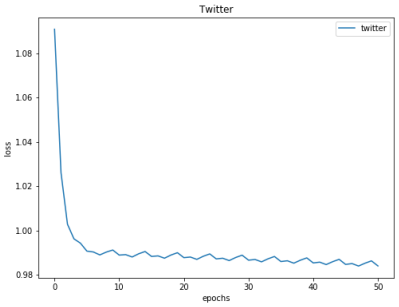

Figure 39: Loss

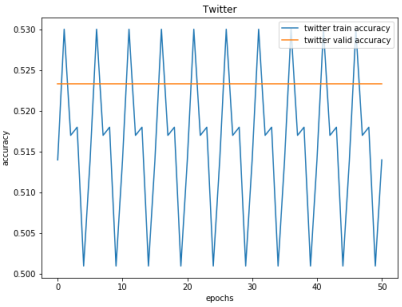

Figure 40: Accuracy

4.3 Dolphins Dataset

Following plot shows accuracy and loss when neural network was trained on dolphins dataset.

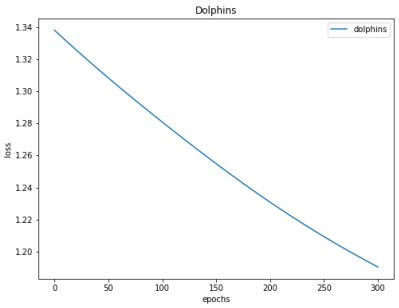

Figure 41: Loss

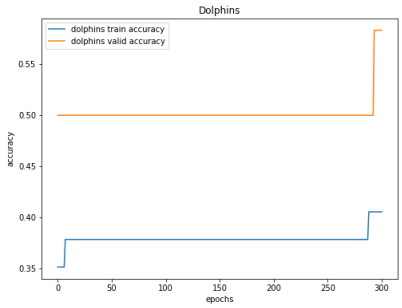

Figure 42: Accuracy