LeNet-5 CNN for MNIST digit classification

CS698U- Computer Vision

Vaibhav Nagar (14785)
Email: vaibhavn@iitk.ac.in

March 19, 2017

1. Implemented LeNet-5 architecture

Architecture of LeNet-5 is modified by using non-linearity functions- Relu and for subsampling, MaxPooling is used.
Conv-Relu, MaxPool, Conv-Relu, MaxPool, FC-Sigmoid, FC-Sigmoid, FC-Softmax

2. Comparison between the time taken by the conv layers vs. the fc layers

Layer	Numpy implementation (ms)	Tensorflow implementation (ms)
Conv-1	0.803	0.303
MaxPool-1	0.105	0.025
Conv-2	2.435	0.169
MaxPool-2	0.0656	0.009
FC-1	0.235	0.05
FC-2	0.0806	0.014
FC-3 (Softmax)	0.032	0.016

Table 1: Feedforward time calculated for each layer on one image.

- Time Taken by conv layers = 3.238 ms on Numpy, 0.472 ms on Tensorflow
- Time taken by fc layers = 0.348 ms on Numpy, 0.08 ms on Tensorflow
- Total FeedForward Time for an image = 3.757 ms on Numpy, 0.601 ms on Tensorflow

3. Comparison between number of params in the conv layers vs. the fc layers

- Conv Layer-1: 156 parameters
- Max Pool-1: 0
- Conv Layer-2: 2416 parameters
- Max Pool-2: 0
- FC Layer-1: 48120 parameters
- FC Layer-2: 10164 parameters
- FC Layer-3: 850 parameters

Total conv layers parameters = 2572 parameters
Total fc layers parameters = 59134 parameters
Total parameters in LeNet-5 = 61706 parameters

4. Plots of training and validation error rates vs. the number of iterations

- Training Set of 50,000 images, validation set of 10,000 images and test set of 10,000 images are used for training and testing.
- Three Implementations:

Implementation	Architecture	Gradient Optimizer	Epochs	Learning Rate	L2 Regularization
Numpy	LeNet-5	Adam	4	0.001	0
Tensorflow	LeNet-5	Adam	5	0.001	0
MLP	2 hidden layers (120 and 60 nodes)	AdaGrad	5	0.01	0

4.1 Training and Test dataset Accuracy (Best)

Batch Size	Numpy (Training set)	Numpy (Test set)	Tensorflow (Test set)	MLP (Test set)
16	99.1%	98.54%	98.99%	95.09%
32	98.96%	98.55%	99.1%	94.64%
64	98.74%	98.57%	99.2%	93.86%
128	98.35%	98.3%	98.9%	93.1%

Table 2: Training and Test accuracy on different batch sizes

4.2 Batch Size = 16

- Numpy Implementation learning curves (epochs=4):

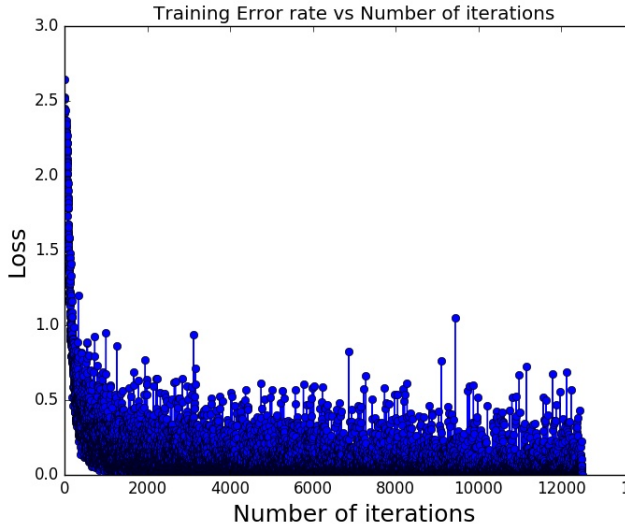

Figure 1: Training loss on batch size = 16 (Numpy)

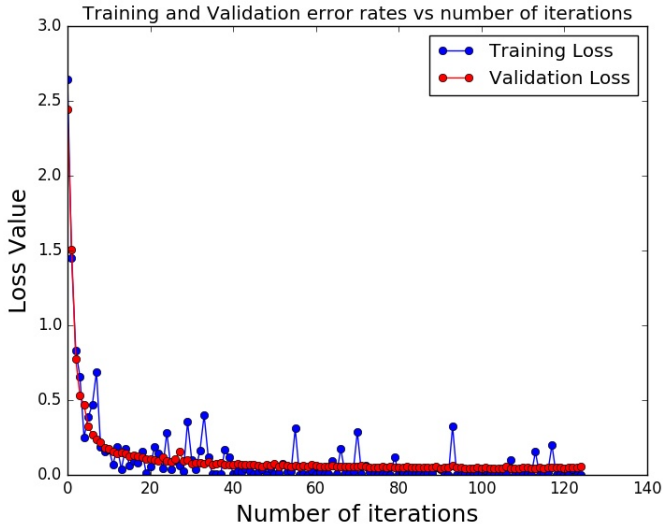

Figure 2: Training and Validation error rates on batch size = 16 (Numpy)

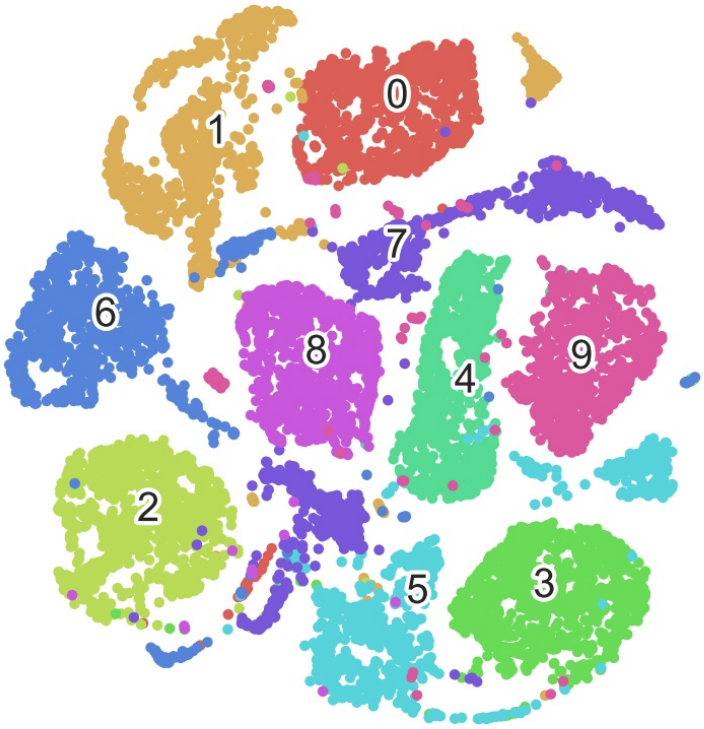

Figure 3: TSNE plot of activations of final fc layer on batch size = 16 (Numpy)

- Tensorflow Implementation learning curves (epochs=4):

Figure 4: Training loss on batch size = 16 (Tensorflow)

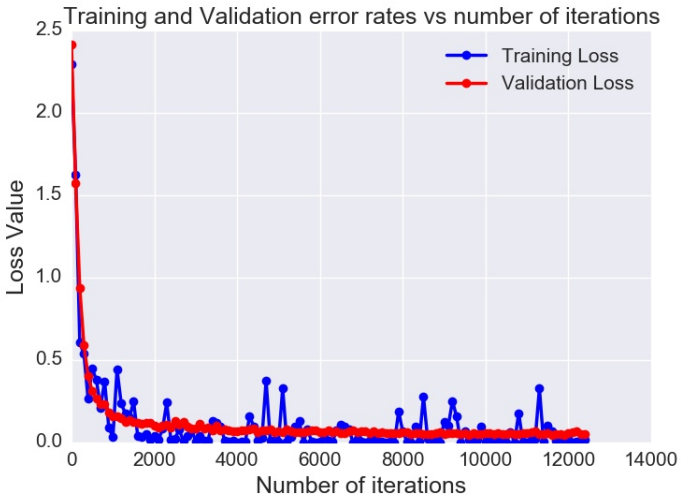

Figure 5: Training and Validation error rates on batch size = 16 (Tensorflow)

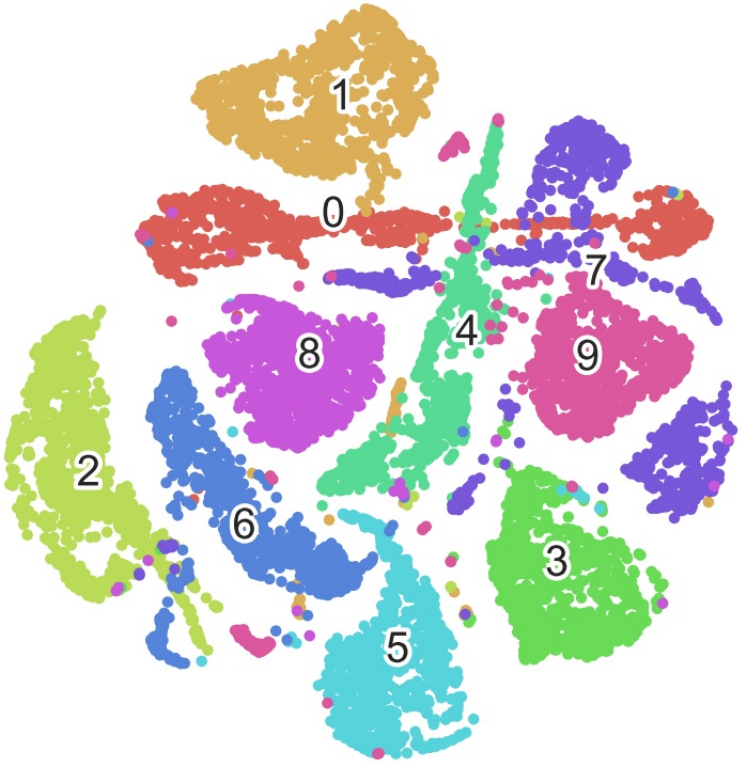

Figure 6: TSNE plot of activations of final fc layer on batch size = 16 (Tensorflow)

- MLP (Two hidden layers (120 and 60 nodes) with tanh non-linearity) learning curves (epochs=5, method=adagrad):

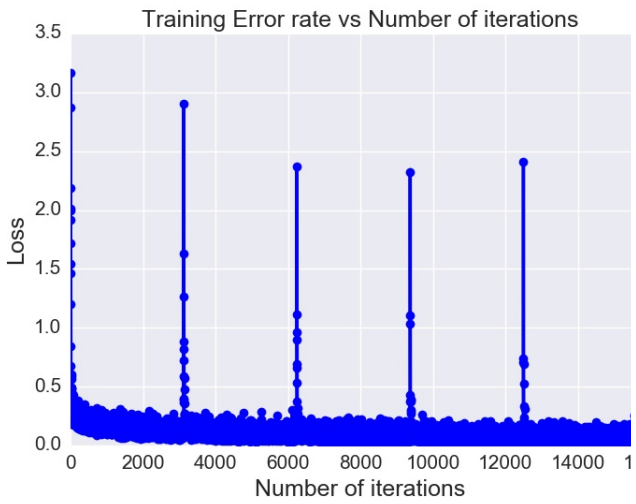

Figure 7: Training loss on batch size = 16 (MLP)

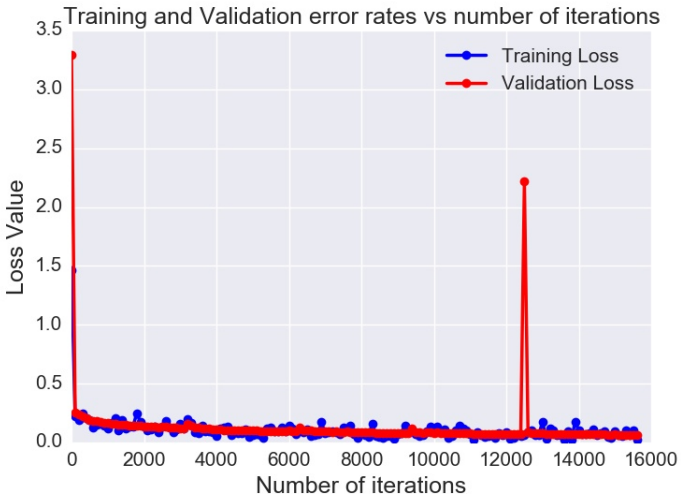

Figure 8: Training and Validation error rates on batch size = 16 (MLP)

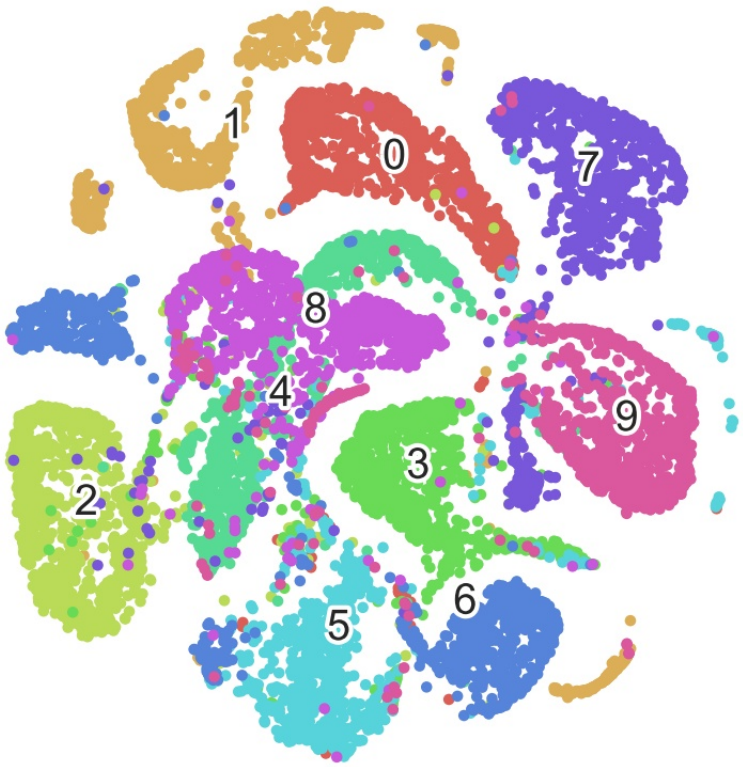

Figure 9: TSNE plot of activations of final fc layer on batch size = 16 (MLP)

4.3 Batch Size = 32

- Numpy Implementation learning curves (epochs=4):

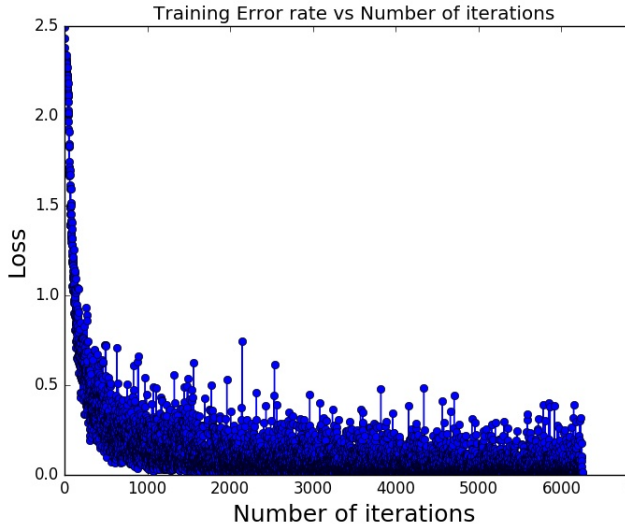

Figure 10: Training loss on batch size = 32 (Numpy)

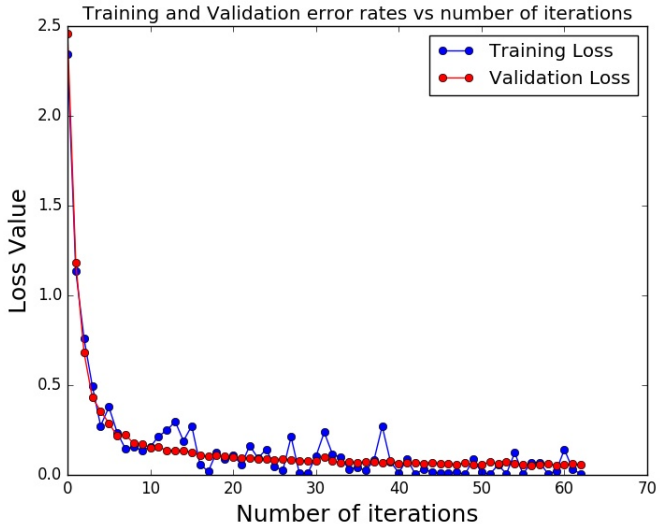

Figure 11: Training and Validation error rates on batch size = 32 (Numpy)

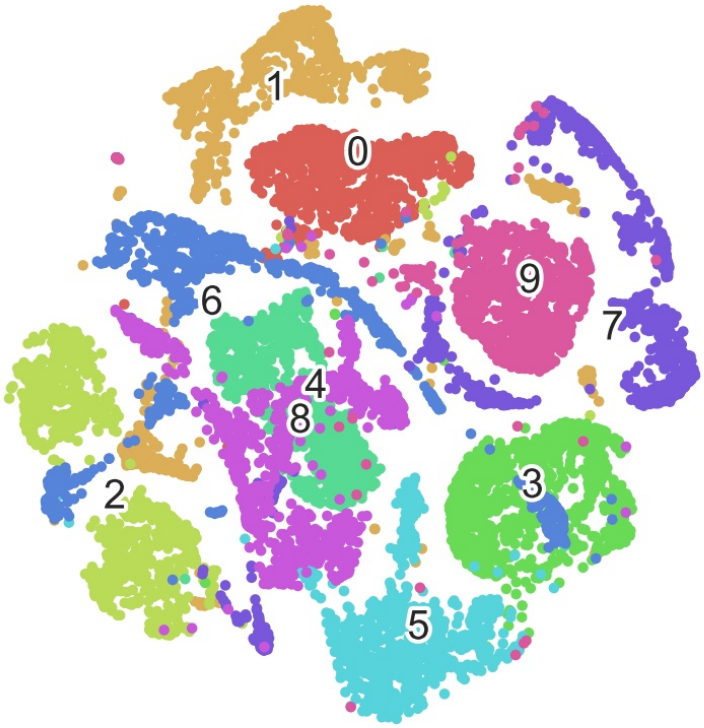

Figure 12: TSNE plot of activations of final fc layer on batch size = 32 (Numpy)

- Tensorflow Implementation learning curves (epochs=4):

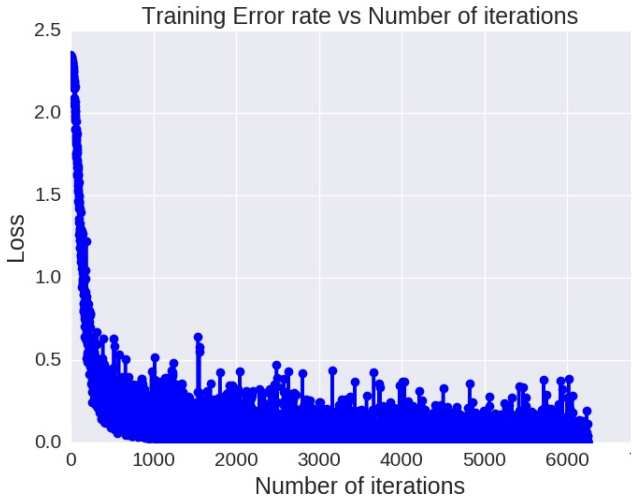

Figure 13: Training loss on batch size = 32 (Tensorflow)

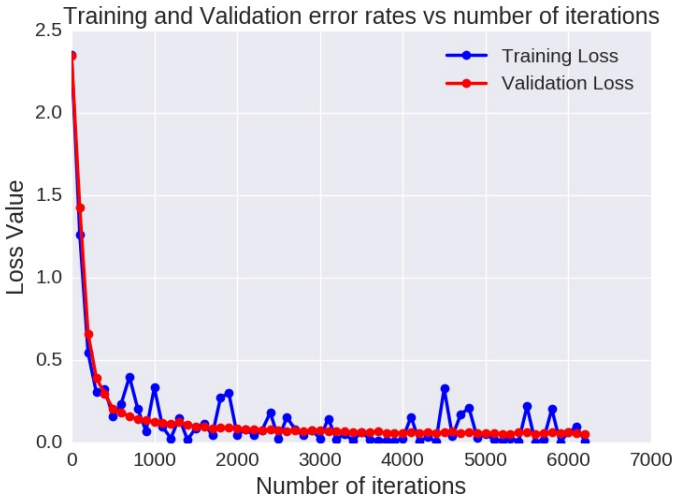

Figure 14: Training and Validation error rates on batch size = 32 (Tensorflow)

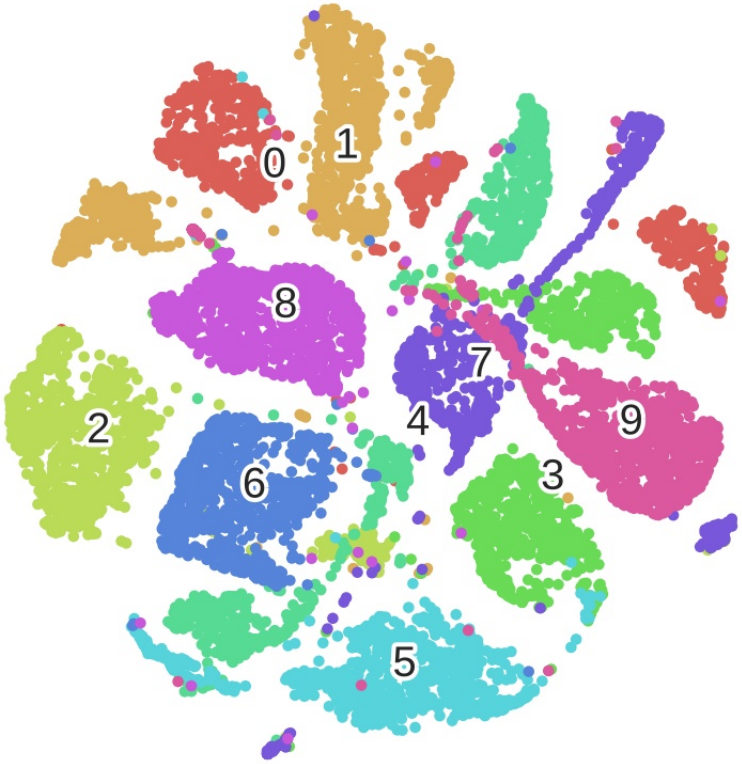

Figure 15: TSNE plot of activations of final fc layer on batch size = 32 (Tensorflow)

- MLP (Two hidden layers (120 and 60 nodes) with tanh non-linearity) learning curves (epochs=5, method=adagrad):

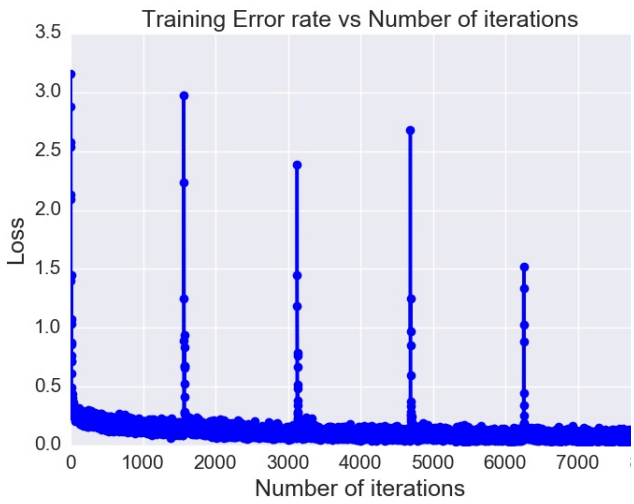

Figure 16: Training loss on batch size = 32 (MLP)

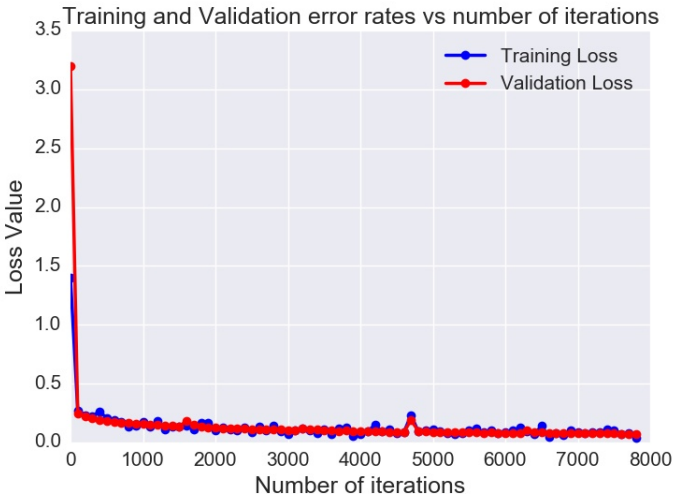

Figure 17: Training and Validation error rates on batch size = 32 (MLP)

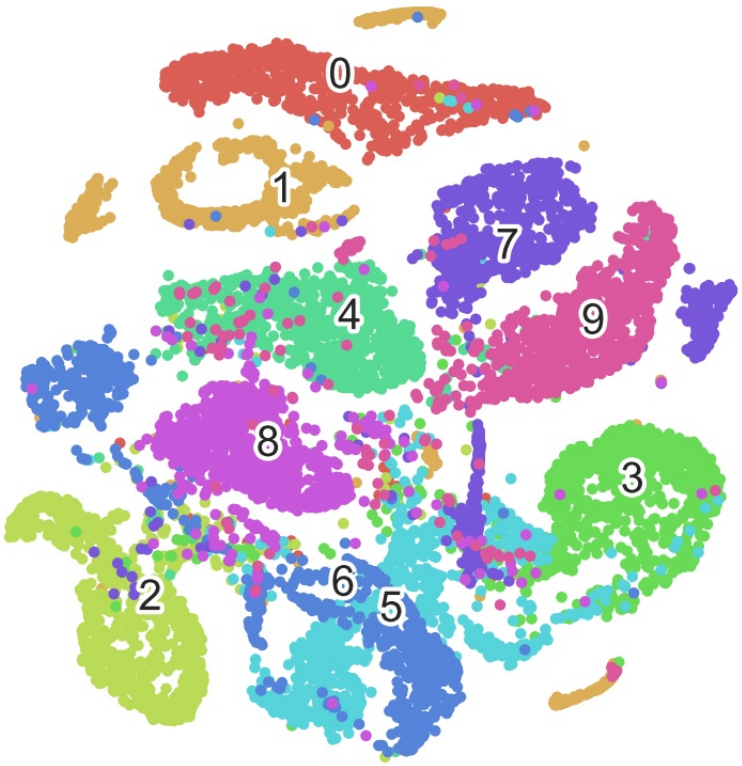

Figure 18: TSNE plot of activations of final fc layer on batch size = 32 (MLP)

4.4 Batch Size = 64

- Numpy Implementation learning curves (epochs=4):

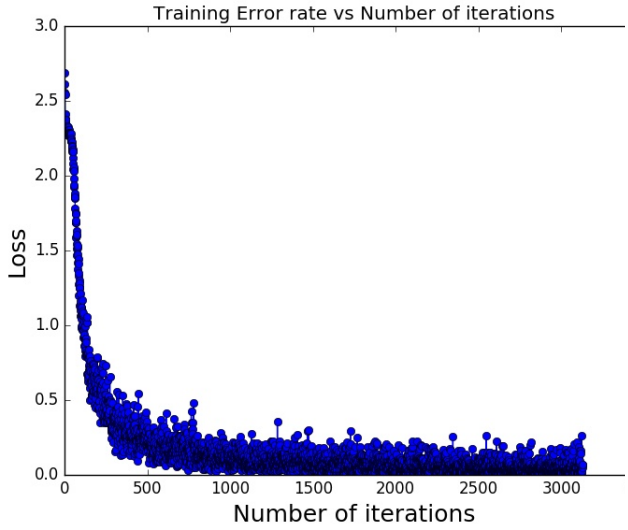

Figure 19: Training loss on batch size = 64

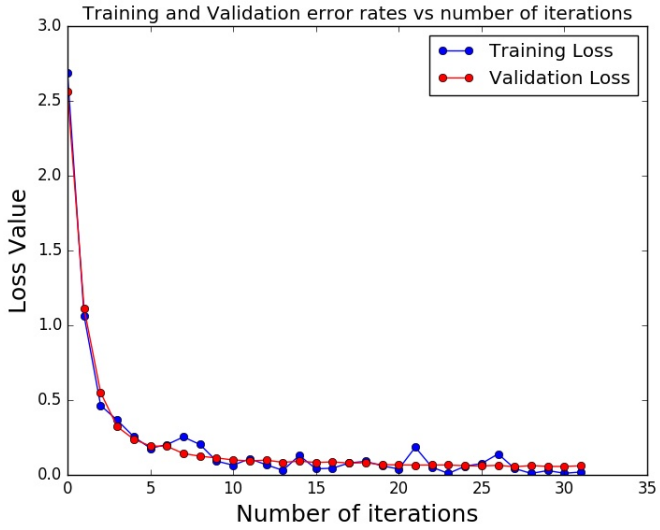

Figure 20: Training and Validation error rates on batch size = 64

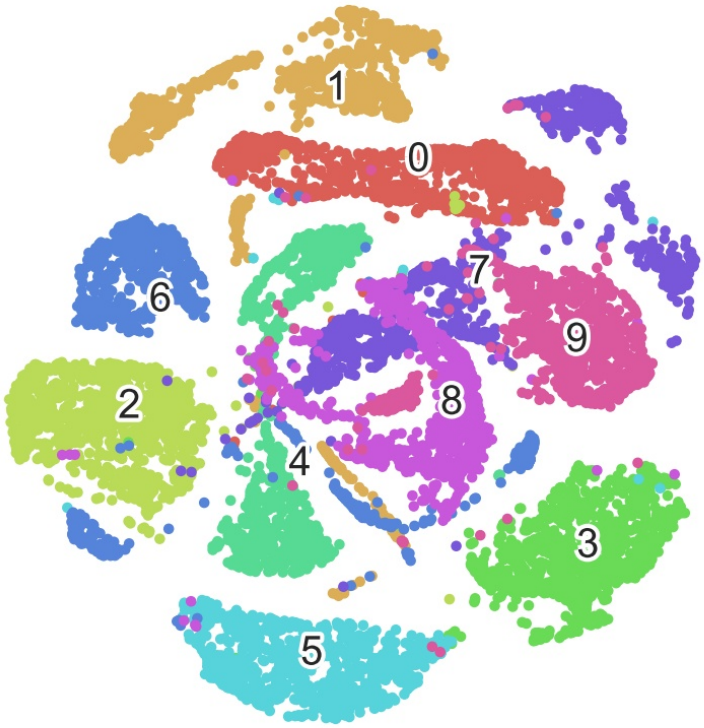

Figure 21: TSNE plot of activations of final fc layer on batch size = 64

- Tensorflow Implementation learning curves (epochs=4)

Figure 22: Training loss on batch size = 64

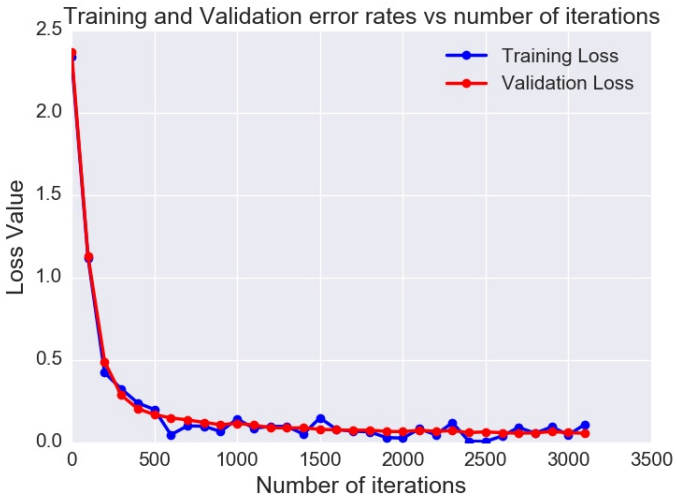

Figure 23: Training and Validation error rates on batch size = 64

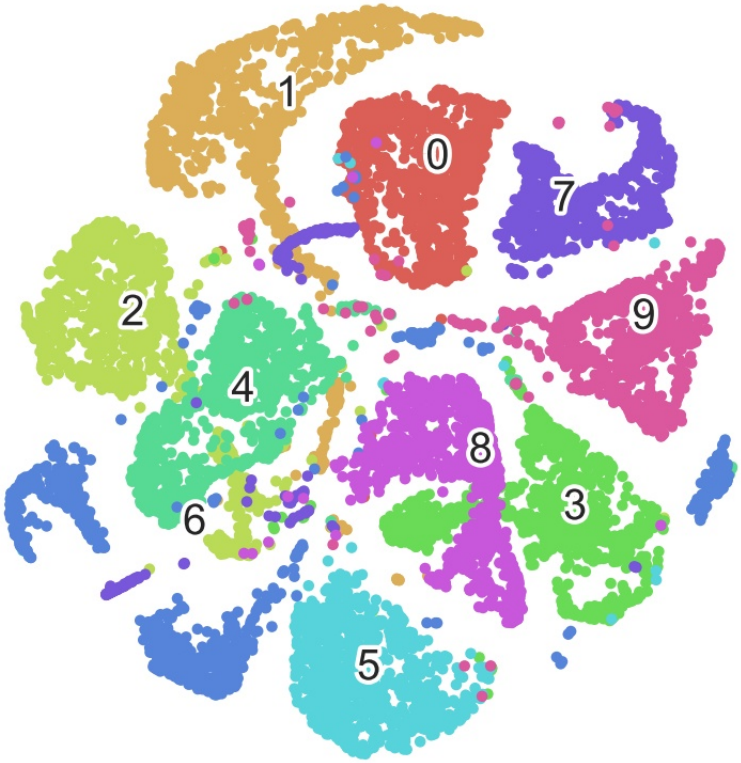

Figure 24: TSNE plot of activations of final fc layer on batch size = 64

- MLP (Two hidden layers (120 and 60 nodes) with tanh non-linearity) learning curves (epochs=5, method=adagrad):

Figure 25: Training loss on batch size = 64 (MLP)

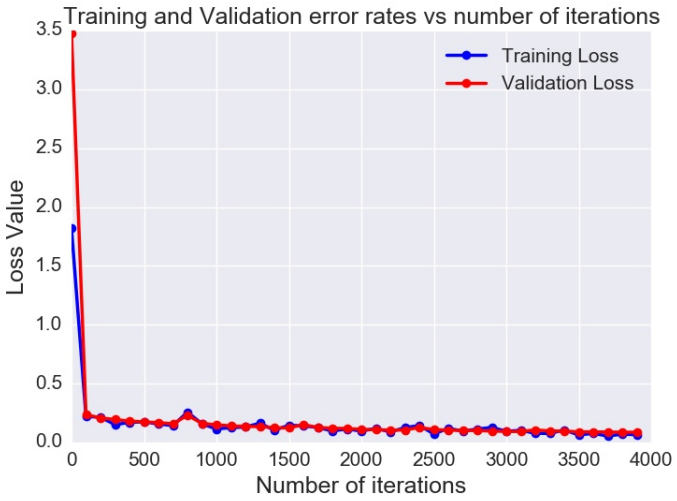

Figure 26: Training and Validation error rates on batch size = 64 (MLP)

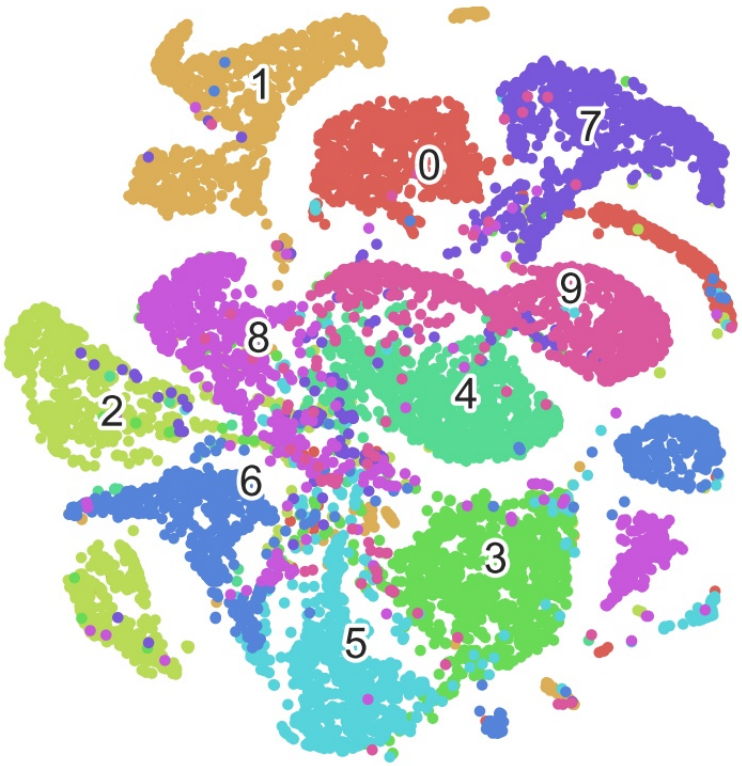

Figure 27: TSNE plot of activations of final fc layer on batch size = 64 (MLP)

4.5 Batch Size = 128

- Numpy Implementation learning curves (epochs=4):

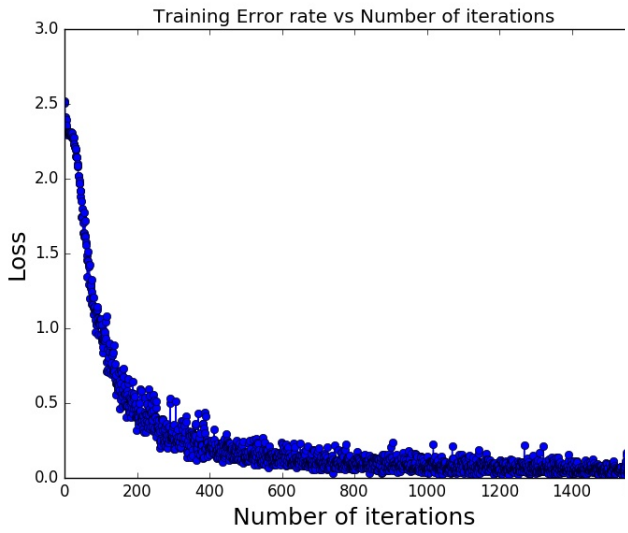

Figure 28: Training loss on batch size = 128

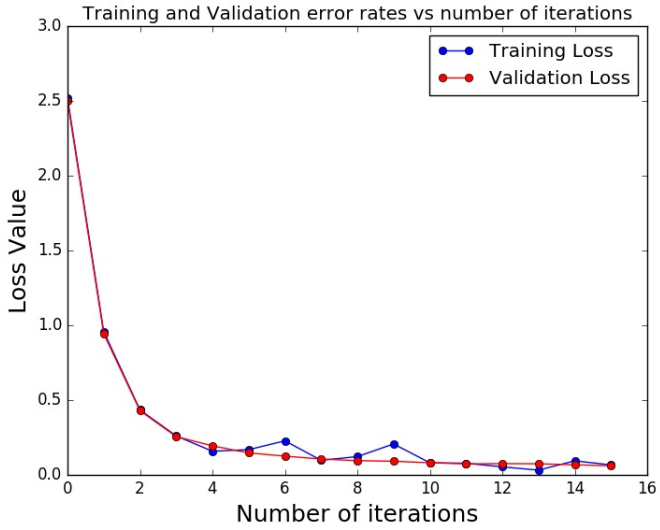

Figure 29: Training and Validation error rates on batch size = 128

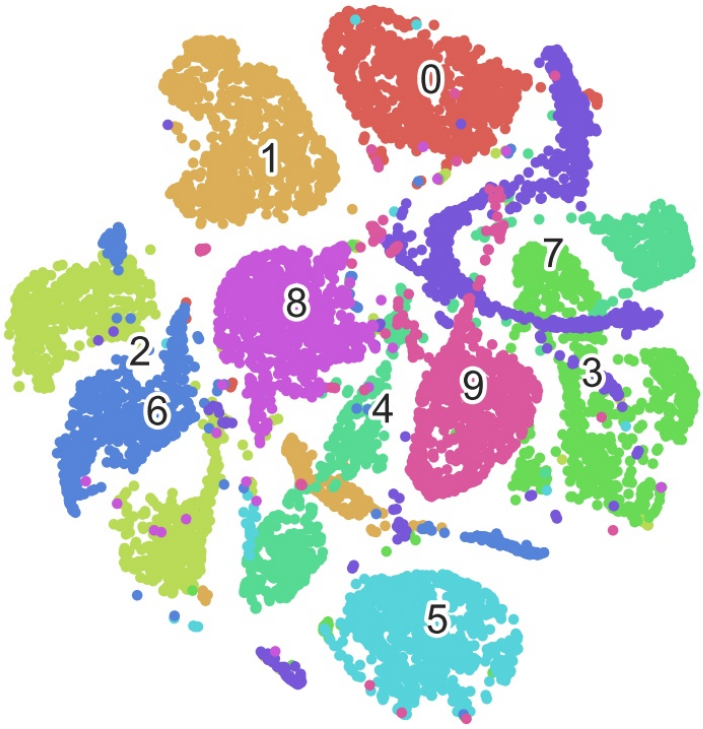

Figure 30: TSNE plot of activations of final fc layer on batch size = 128

- Tensorflow Implementation learning curves (epochs=4):

Figure 31: Training loss on batch size = 128

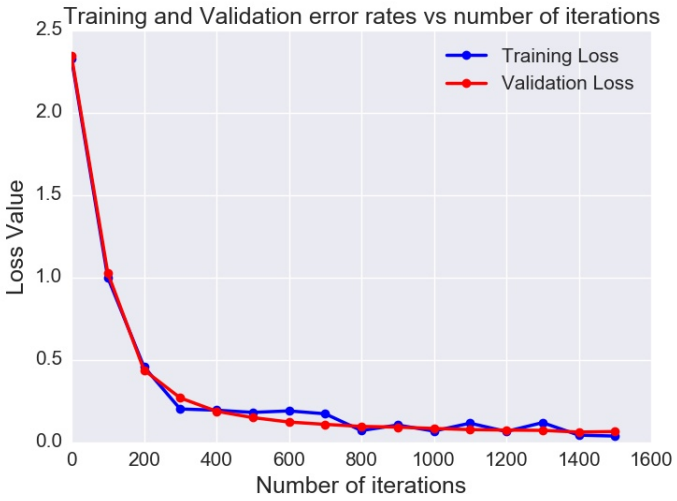

Figure 32: Training and Validation error rates on batch size = 128

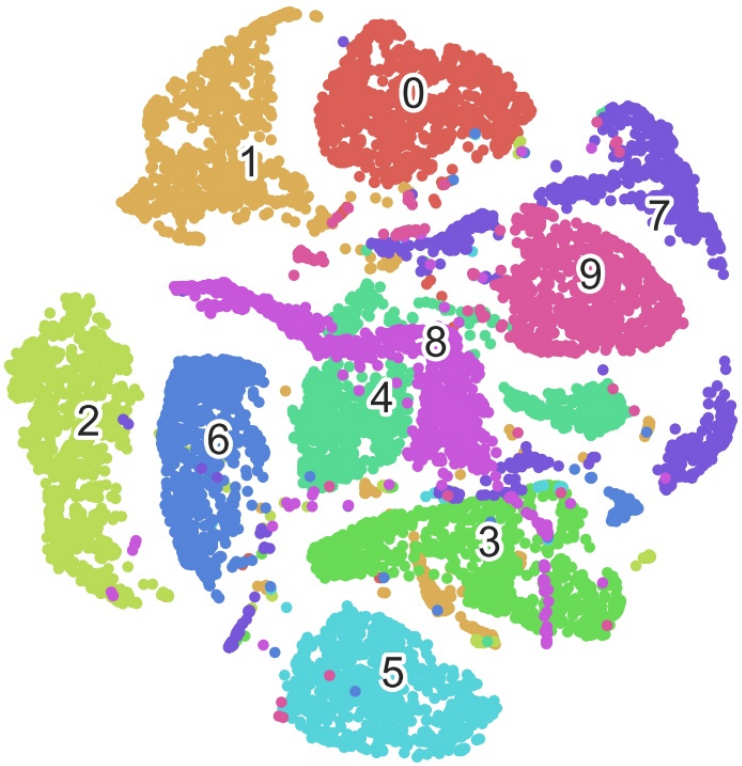

Figure 33: TSNE plot of activations of final fc layer on batch size = 128

- MLP (Two hidden layers (120 and 60 nodes) with tanh non-linearity) learning curves (epochs=5, method=adagrad):

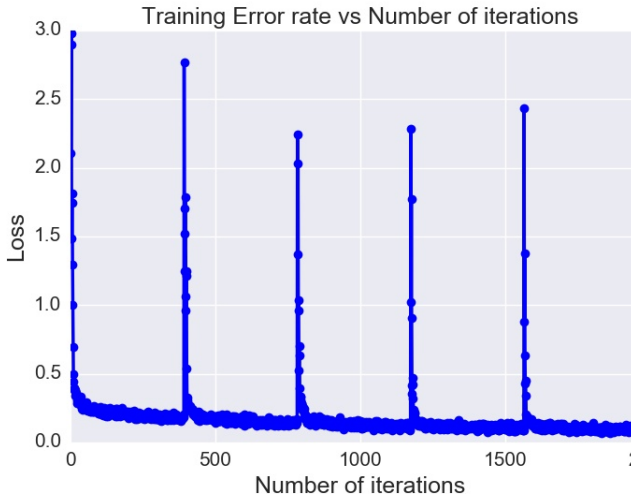

Figure 34: Training loss on batch size = 128 (MLP)

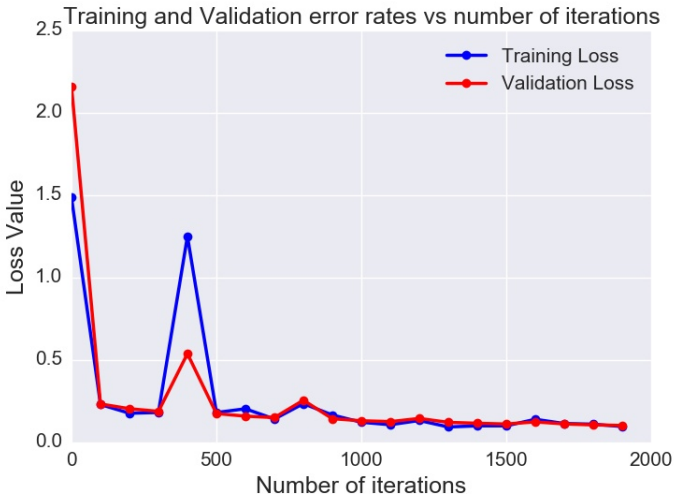

Figure 35: Training and Validation error rates on batch size = 128 (MLP)

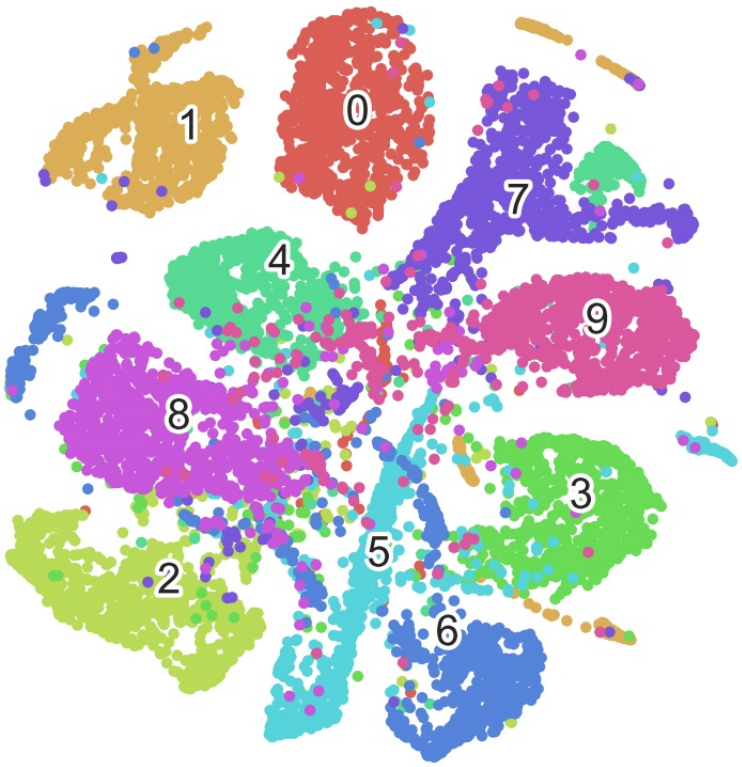

Figure 36: TSNE plot of activations of final fc layer on batch size = 128 (MLP)

4.6 Feature Maps of conv and maxpool layers for each digit on different batch sizes

These Feature maps are present in "visualize_feature_maps" folder for each digit.

5. Inferences

- For batch size=64 , highest test accuracy is achieved after training for 4 epochs.
- L2 regularization with (0.01 regularization parameter), gives slightly less accuracy than without regularization when trained for 4 epochs. With regularization, network needs to be trained for more number of epochs.
- Training and feedforward time in tensorflow implementation is much lesser than numpy, as it is highly optimized.