

# HW 3

---

Jinrui Hou

<https://github.com/kinreehou/caltech-ee148-spring2020-hw03>

## 1. Augmentation scheme

Perform random rotation on each picture.

I used the provided fcNet to compare the results after 5 epochs with and without the data augmentation. I test different max degree value and the results are shown below.

```
transforms.RandomRotation(max_degree)      # randomly rotate  
(-max_degree, max_degree)
```

Augmentation	Accuracy
No augmentation	44%
Augmentation max_degree 8	46%
Augmentation max_degree 15	52%
Augmentation max_degree 30	42%

The results shows that the data augmentation scheme works, but only when the parameters are set properly. Because all the MNIST data are basically processed and the handwritten numbers are not that oblique.

## 2. Process to develop the architecture

First start from models contains only fully connected layers, dropout layers and reLUs. The results are not very good. Then I started to try with the convolution layers. When the whole structure was completed(number of layers and the order of different layers), I trained the model with different parameters(the size of each layer and the dropout rate). When the drop rate is too high, the model is ineffective.

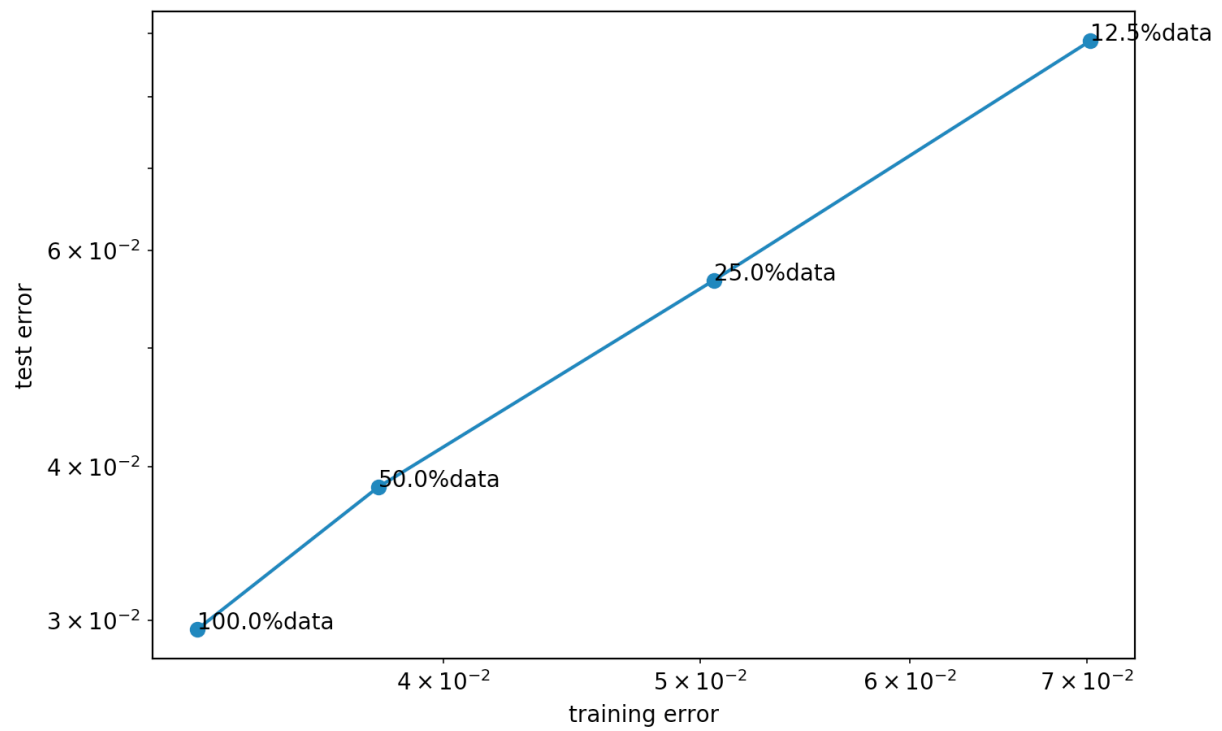
My best architecture is:

```
torch.Size([6, 1, 5, 5])
torch.Size([6])
torch.Size([6])
torch.Size([6])
torch.Size([16, 6, 5, 5])
torch.Size([16])
torch.Size([16])
torch.Size([16])
torch.Size([120, 256])
torch.Size([120])
torch.Size([10, 120])
torch.Size([10])
total params: 34666
```

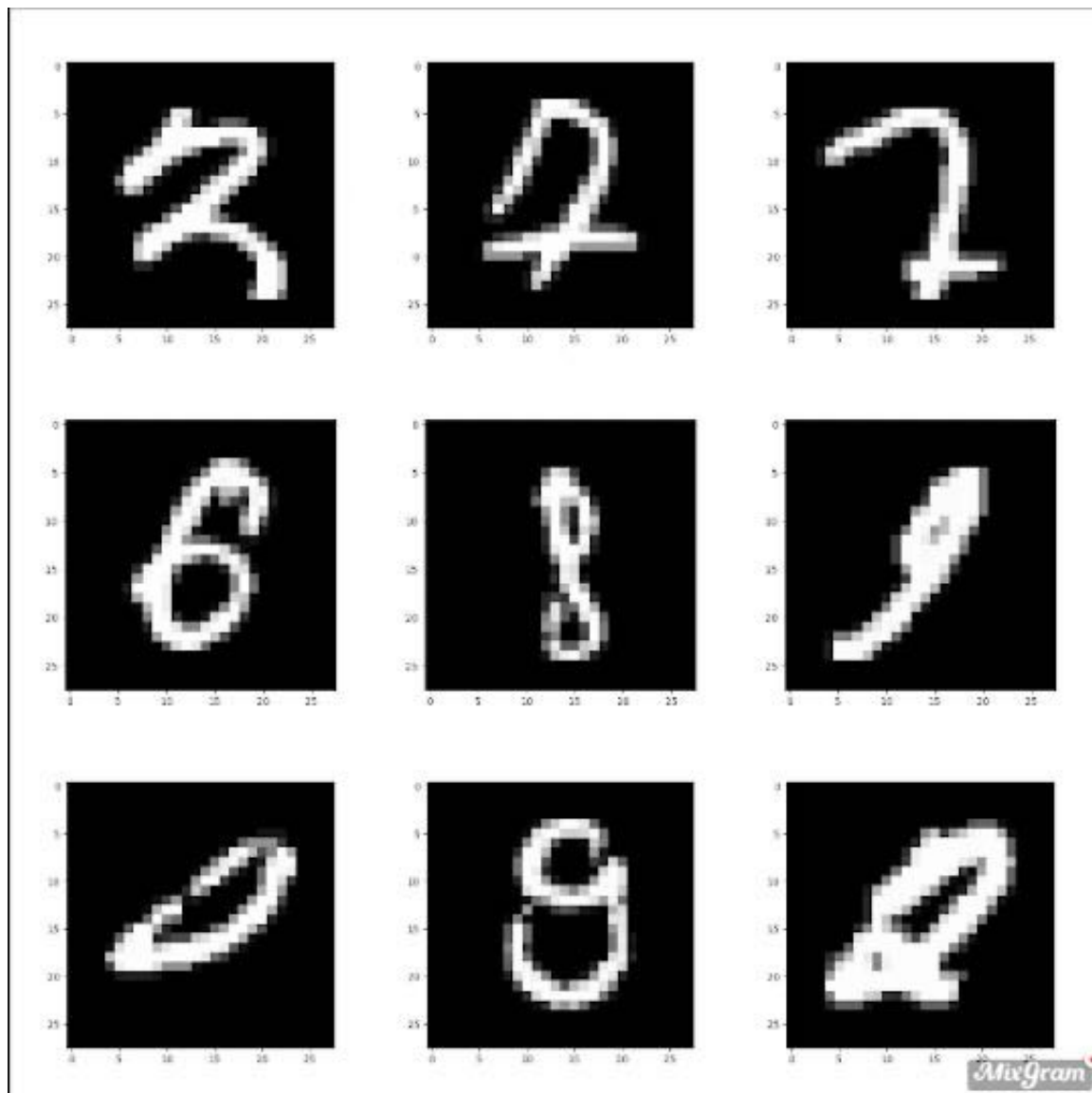
3.

Training Data	Training Error	Val Error	Val Accuracy	Test error	Test Accuracy
all data	0.0323	0.0381	99%	0.0295	99%
1/2 data	0.0378	0.0504	98%	0.0385	99%
1/4 data	0.0506	0.0702	98%	0.0567	98%
1/8 data	0.0702	0.1533	96%	0.0888	97%

Log-log figure:



#### 4. Mistakes made by my classifier



Those images are really hard to classify even by human eyes.

## 5. Learned kernel visualization

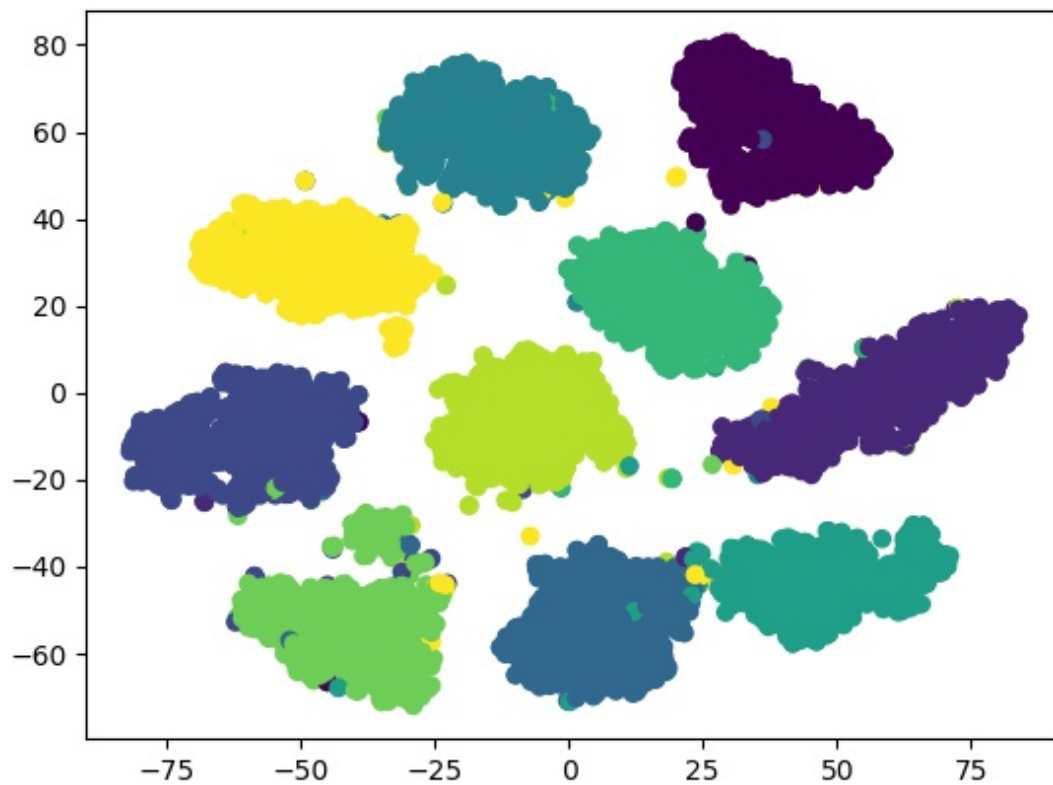
I only have 6 feature maps for the first convolution layer.

## 6. Confusion Matrix





































```
[[ 977.  0.  0.  0.  0.  0.  1.  1.  1.  0.]
 [ 0.1132.  1.  1.  0.  1.  0.  0.  0.  0.  0.]
 [ 1.  2.1024.  0.  0.  0.  0.  0.  4.  1.  0.]
 [ 0.  0.  2.999.  0.  5.  0.  3.  1.  0.]
```

```
[ 0.  0.  0.  0. 978.  0.  1.  0.  1.  2.]
[ 2.  0.  0.  6.  0. 882.  1.  1.  0.  0.]
[ 5.  2.  0.  0.  2.  6. 943.  0.  0.  0.]
[ 0.  3.  5.  0.  0.  0.  0.1017.  1.  2.]
[ 4.  0.  1.  1.  2.  0.  0.  4. 959.  3.]
[ 1.  2.  0.  0.  4.  3.  0.  3.  0. 996.]]
```

## 7. tSNE visualization



## 8. Nearest images

I0	I1	I2	I3	I4	I5	I6	I7	I8
								
								
								
								

The nearest images to I0 do not show the same digit as I0 does. This may reveal that the network is not linear and Euclidean distance cannot represents each digit class.