ECE 544NA Fall 2016 Assignment 2

Jifu Zhao Date: 09/29/2016

I. Pencil-and-Paper

1. Derivative of Softmax
Recall the softmax function:

$$\vec{y}[k] = \frac{e^{\vec{z}[k]}}{\sum_{l=1}^{C} e^{\vec{z}[l]}} \tag{1}$$

So, consider two cases, j = k and j != k.

When j = k:

$$\frac{\partial \vec{y}[k]}{\partial \vec{z}[j]} = -\frac{e^{\vec{z}[k]} \cdot \sum_{l=1}^{C} e^{\vec{z}[l]} - e^{\vec{z}[k]} \cdot e^{\vec{z}[k]}}{(\sum_{l=1}^{C} e^{\vec{z}[l]})^2} = \vec{y}[k] - \vec{y}[k] \cdot \vec{y}[k] \tag{2}$$

When j != k:

$$\frac{\partial \vec{y}[k]}{\partial \vec{z}[j]} = -\frac{e^{\vec{z}[k]} \cdot e^{\vec{z}[j]}}{(\sum_{l=1}^{C} e^{\vec{z}[l]})^2} = -\vec{y}[k] \cdot \vec{y}[j] \tag{3}$$

So, we can conclude that:

$$\frac{\partial \vec{y}[k]}{\partial \vec{z}[j]} = \begin{cases} -\vec{y}[k] \cdot \vec{y}[j] & \text{if } k! = j \\ \vec{y}[k] - \vec{y}[k] \cdot \vec{y}[k] & \text{if } k = j \end{cases} \tag{4}$$

2. Negative Log Likelihood loss for Multi-Class.
Recall the negative log likelihood:

$$L = -\sum_{i}^{N} \sum_{k}^{K} \mathbf{1}[y_i = k] \cdot log(\hat{\vec{y}}_i[k]) \tag{5}$$

In this way, we have:

$$\frac{\partial L}{\partial \hat{\vec{y}}_i[j]} = -\mathbf{1}[y_i = k] \cdot \frac{\partial log(\hat{\vec{y}}_i[j])}{\partial \hat{\vec{y}}_i[j]} = -\frac{\mathbf{1}[y_i = j]}{\hat{\vec{y}}_i[j]} \tag{6}$$

3. Avg-pooling (1D)
Recall Avg-pooling (1D) operation with window size W:

$$\vec{y}[i] = \frac{1}{W} \sum_{j=0}^{W} \vec{x}[i+j] \tag{7}$$

Then, we have:

$$\frac{\partial \vec{y}[i]}{\partial \vec{x}[j]} = \begin{cases} -\frac{1}{W} & \text{if } i \leq j \leq i + W \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

4. Max-pooling (1D)
Recall Max-pooling (1D) operation with window size W:

$$\vec{y}[i] = \max_{j=0}^{W} \vec{x}[i+j] \tag{9}$$

Then, we have:

$$\frac{\partial \vec{y}[i]}{\partial \vec{x}[j]} = \begin{cases} 1 & \text{if } \max_{k=0}^{W} \vec{x}[i+k] = \vec{x}[j] \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

5. Convolutional layer (1D)
Recall Convolution (1D) operation, assume $\vec{w}$ is length 3, and zero index at the center:

$$\vec{y}[i] = (\vec{w} * \vec{x})[i] = \sum_{j=-1}^{1} \vec{x}[i-j] \cdot \vec{x}[j] \tag{11}$$

From above equation, we have:

$$\frac{\vec{y}[i]}{\vec{x}[j]} = \begin{cases} \vec{w}[i-j] & \text{if } i-1 \le j \le i+1 \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

$$\frac{\vec{y}[i]}{\vec{w}[j]} = \begin{cases} \vec{x}[i-j] & \text{if } j = -1 or 0 or 1 \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

II. Code-from-Scratch

1. Methods

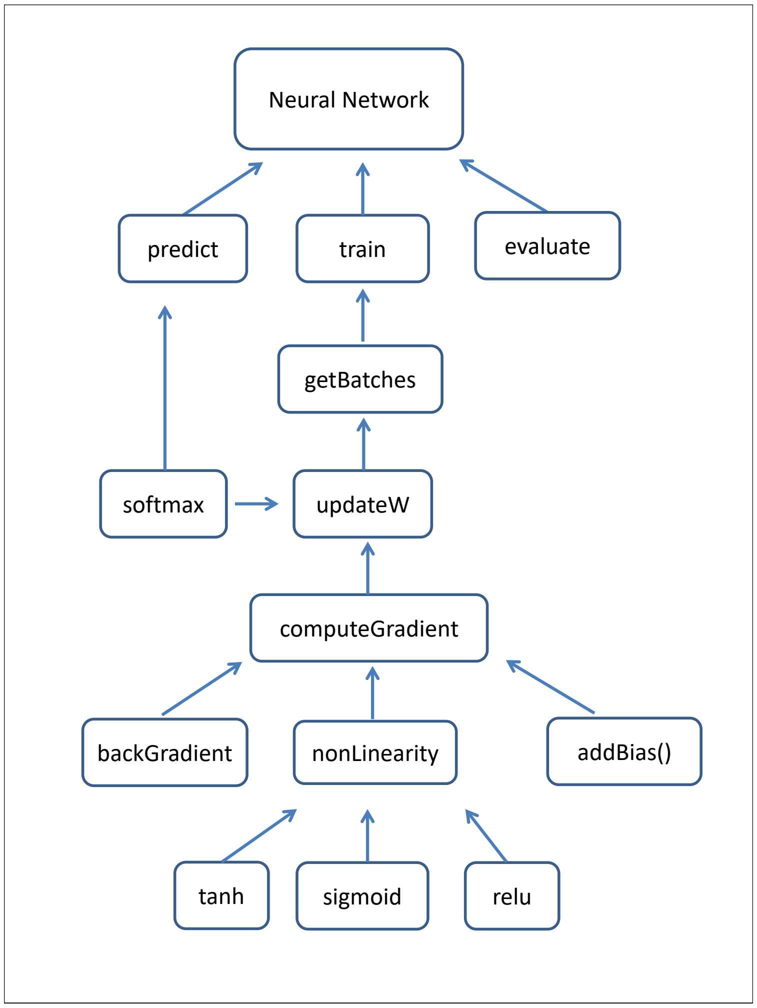


Figure 1: Algorithm Structure

2. Results

Table 1: Comparison of training accuracy

Hidden Nodes/Function	ReLU	Sigmoid	Tanh
10	66.42%	57.25%	63.16%
20	87.94%	76.72%	59.21%
30	60.71%	78.08%	75.35%
40	56.84%	83.69%	77.91%
50	69.90%	83.67%	79.72%

Table 2: Comparison of test accuracy

Hidden Nodes/Function	ReLU	Sigmoid	Tanh
10	39.38%	43.01%	41.03%
20	43.34%	46.09%	44.44%
30	43.23%	44.44%	44.11%
40	35.42%	45.10%	46.09%
50	40.70%	46.31%	43.01%

Table 3: Average time for one iteration (in seconds)

Hidden Nodes/Function	ReLU	Sigmoid	Tanh
10	0.00067	0.00309	0.00509
20	0.00073	0.00330	0.00535
30	0.00082	0.00352	0.00555
40	0.00092	0.00366	0.00582
50	0.00105	0.00383	0.00614

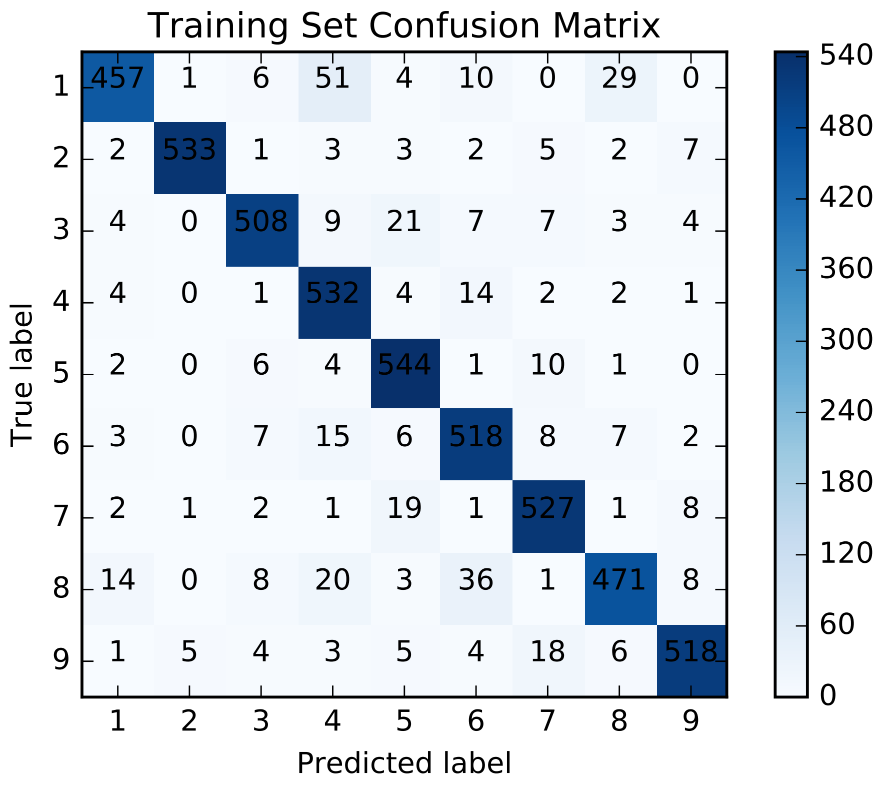
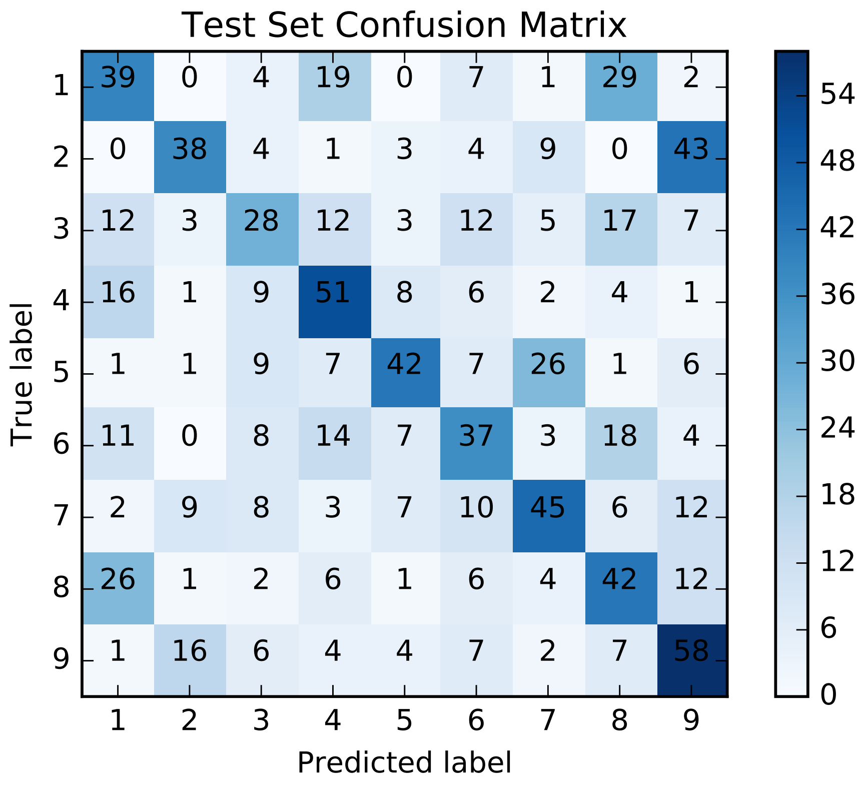


Figure 2: Training Set Confusion Matrix



True label \ Predicted label	1	2	3	4	5	6	7	8	9
1	39	0	4	19	0	7	1	29	2
2	0	38	4	1	3	4	9	0	43
3	12	3	28	12	3	12	5	17	7
4	16	1	9	51	8	6	2	4	1
5	1	1	9	7	42	7	26	1	6
6	11	0	8	14	7	37	3	18	4
7	2	9	8	3	7	10	45	6	12
8	26	1	2	6	1	6	4	42	12
9	1	16	6	4	4	7	2	7	58

Figure 3: Test Set Confusion Matrix

III. TensorFlow

1. Methods

2. Results

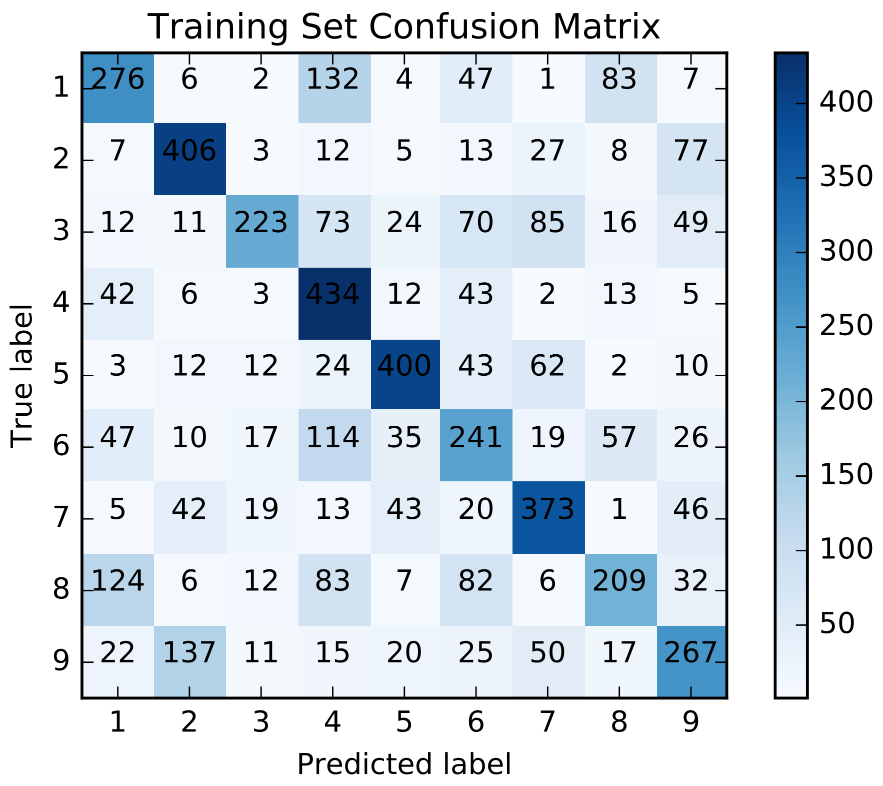
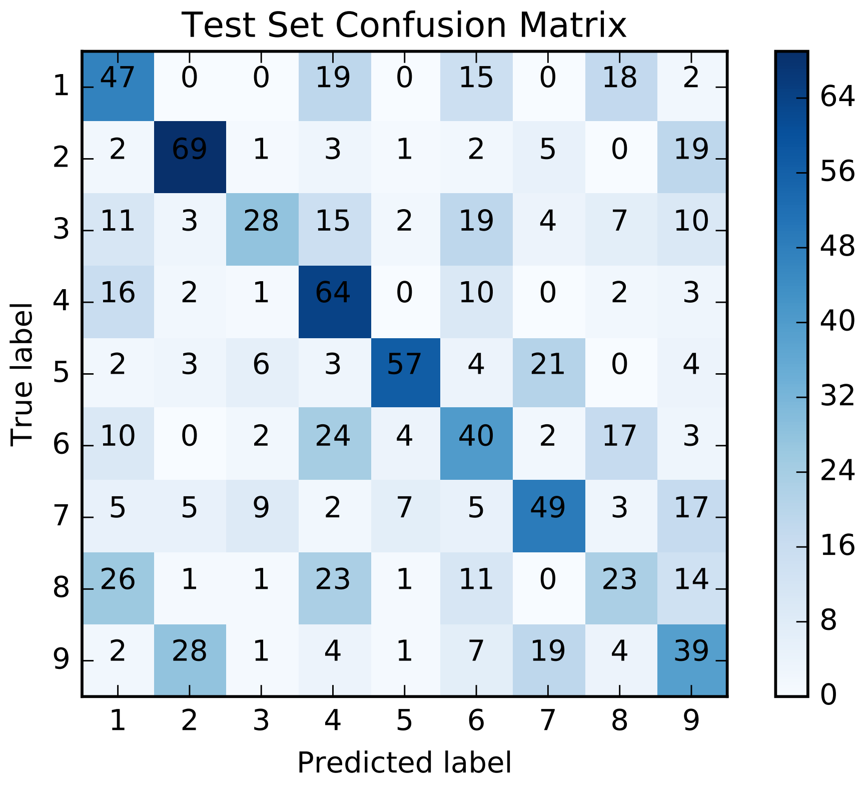


Figure 4: Training Set Confusion Matrix for TDNN model



True label \ Predicted label	1	2	3	4	5	6	7	8	9
1	47	0	0	19	0	15	0	18	2
2	2	69	1	3	1	2	5	0	19
3	11	3	28	15	2	19	4	7	10
4	16	2	1	64	0	10	0	2	3
5	2	3	6	3	57	4	21	0	4
6	10	0	2	24	4	40	2	17	3
7	5	5	9	2	7	5	49	3	17
8	26	1	1	23	1	11	0	23	14
9	2	28	1	4	1	7	19	4	39

Figure 5: Test Set Confusion Matrix for TDNN model

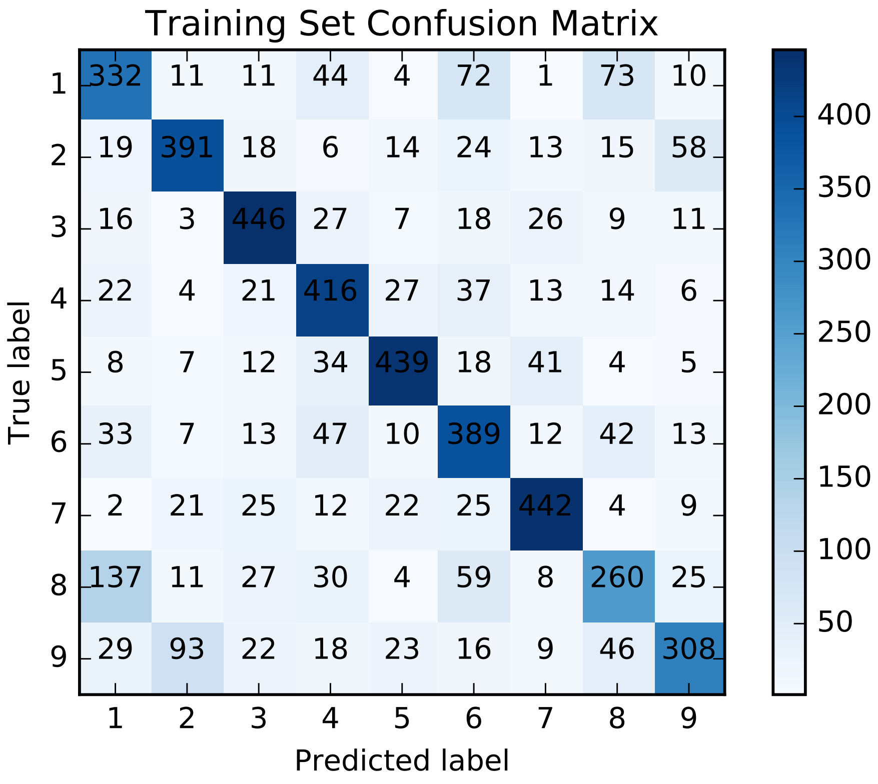


Figure 6: Training Set Confusion Matrix for best model

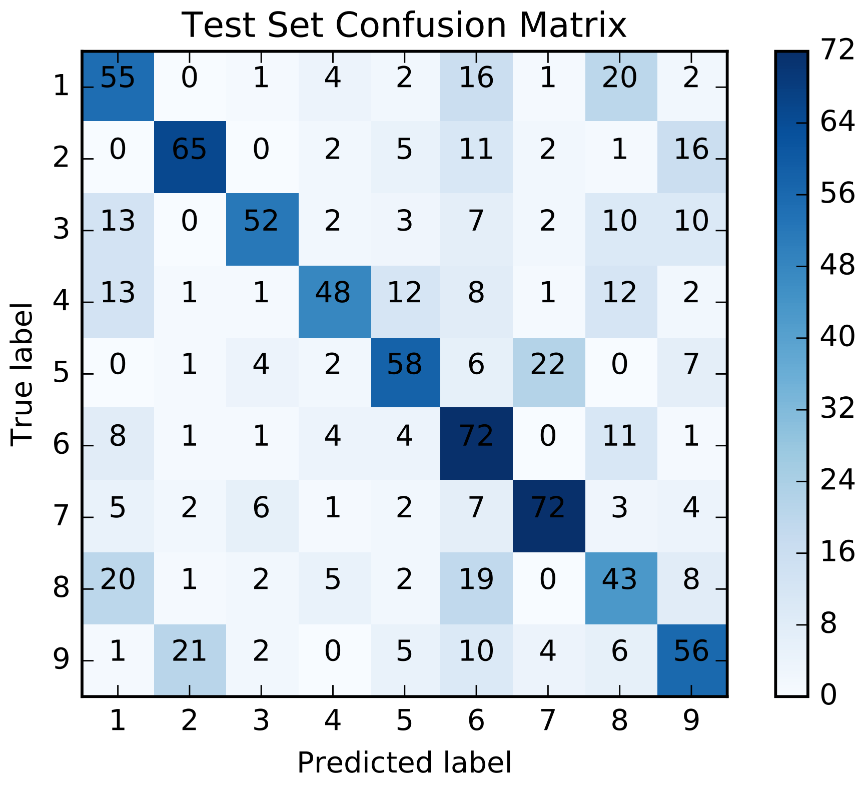


Figure 7: Test Set Confusion Matrix for best model