Deep Learning Assignment 1

Yun-shao Sung* and Chung-Ling Yao†

Abstract. This is the report for deep learning assignment 1

1. Warmup. Write $\frac{\partial E}{\partial X_{in}}$ in terms of $\frac{\partial E}{\partial X_{out}}$

$$\frac{\partial E}{\partial X_{in}} = \frac{\partial E}{\partial X_{out}} \frac{\partial F(X_{in}, W_i)}{\partial X_{in}} = \frac{\partial E}{\partial X_{out}} \frac{e^{X_{in}}}{(1+e^{X_{in}})^2} = \frac{\partial E}{\partial X_{out}} X_{out}(1 - X_{out}) \tag{1.1}$$

2. Multinomial logistic regression. Write the expression of $\frac{\partial (X_{out})_i}{\partial (X_{in})_j}$
if $i = j$, , and let $C = \sum_k e^{(X_I)_k} - e^{(X_I)_i}$

$$(X_o)_i = \frac{e^{(X_I)_i}}{\sum_k e^{(X_I)_k}} = \frac{e^{(X_I)_i}}{e^{(X_I)_0} + e^{(X_I)_i} + \ldots e^{(X_I)_i} + \ldots + e^{(X_I)_k}} = \frac{e^{(X_I)_i}}{C + e^{(X_I)_i}} (2.1)$$

$$\frac{\partial (X_o)_i}{\partial (X_I)_i} = \frac{\partial}{\partial (X_I)_i}\left(\frac{e^{(X_I)_i}}{C + e^{(X_I)_i}}\right) = \frac{-\beta e^{-\beta(X_I)_i}}{C + e^{(X_I)_i}} + \frac{\beta e^{-2\beta(X_I)_i}}{(C + e^{(X_I)_i})^2} = \beta X_o(-1 + X_o)(2.2)$$

if $i \neq j$, and let $K = \sum_k e^{(X_I)_k} - e^{(X_I)_j}$

$$\frac{\partial (X_o)_i}{\partial (X_I)_j} = \frac{\partial}{\partial (X_I)_j}\left(\frac{e^{(X_I)_i}}{K + e^{(X_I)_j}}\right) = \frac{\beta e^{-\beta(X_I)_i} e^{-\beta(X_I)_j}}{(K + e^{(X_I)_i})^2} = \beta (X_o)_i (X_o)_j (2.3)$$

3. Torch (MNIST Handwritten Digit Recognition).

3.1. Experiment.

3.1.1. Original Model. The training and test accuracy of the original model. In the following experiments, we will compare the outcome of different configure with the original model. The training accuracy achieve 100% in epoch 30. The test accuracy shift up and down slightly, but it increase in long term.

*yss265@nyu.edu
†cly264@nyu.edu

Epoch	Training Accuracy (%)	Test Accuracy (%)
1	96.71	99.04
2	99.04	99.41
3	99.34	99.40
4	99.53	99.47
5	99.62	99.48
6	99.77	99.54
7	99.81	99.48
8	99.86	99.43
9	99.89	99.49
10	99.92	99.51

Table 3.1

The training and test accuracy of the original model

3.1.2. Different normalization methods. Use different normalization methods, such as different Gaussian 1D normalization array size or without normalization. Virtualize it to see the effect on the images, and compare the training/test accuracy in the first three epoches.

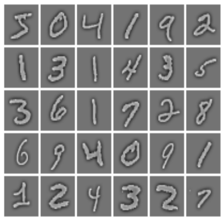
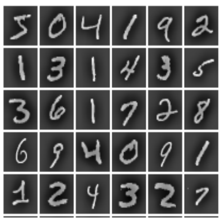
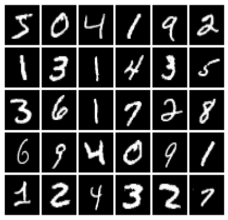
Normalization	Gaussian1D(7)		Gaussian1D(15)		no normalization	
Vitualization						
Accuracy in each epoch	training	test	training	test	training	test
1	96.71	99.04	96.72	98.88	96.66	98.79
2	99.04	99.41	99.00	99.11	98.99	99.14
3	99.34	99.40	99.32	99.28	99.27	99.31

Table 3.2

The training and test accuracy of different normalization methods

3.1.3. Different curve function. The original curve function is tanh. We changed it to be reLU. We found the original tanh function works better in both training and test accuracy.

Loss Function	Tanh(default)		reLU		—	
Accuracy in each epoch	training	test	training	test	training	test
1	96.71	99.04	96.52	98.73	XX.XX	XX.XX
2	99.04	99.41	98.86	99.02	XX.XX	XX.XX
3	99.34	99.40	99.13	99.08	XX.XX	XX.XX

Table 3.3

The training and test accuracy of different curve function

3.1.4. Different loss function. The original loss function is NLL. We changed it to multi-margin and MSE. We find the multi-margin generate the same result with NLL. The MSE looks slightly better in training accuracy, but not in test accuracy.

Loss Function	NLL(default)		Multi-margin		MSE	
Accuracy in each epoch	training	test	training	test	training	test
1	96.71	99.04	96.71	99.04	96.74	99.04
2	99.04	99.41	99.04	99.41	99.07	99.30
3	99.34	99.40	99.34	99.40	99.35	–

Table 3.4

The training and test accuracy of different loss function

3.1.5. Different model structure.