CIS581: Computer Vision and Computational Photography

Project 4: Deep Learning

Jiaxiao Cai
17999524

1. Plot Loss and Gradient (20%)

1.1 Sigmoid Function

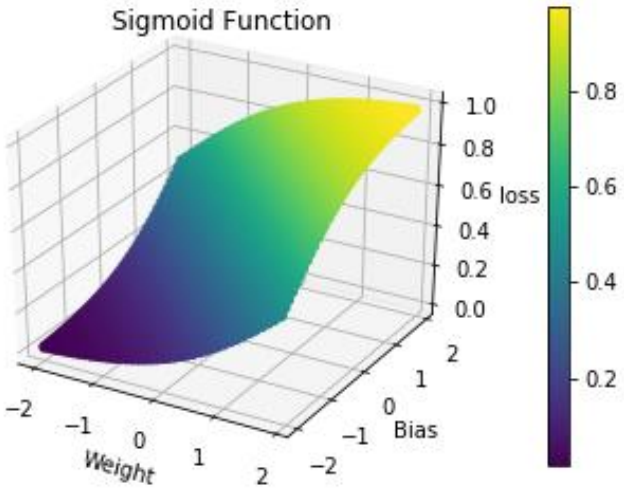

Figure 1 Sigmoid Function

1.2 L2 loss

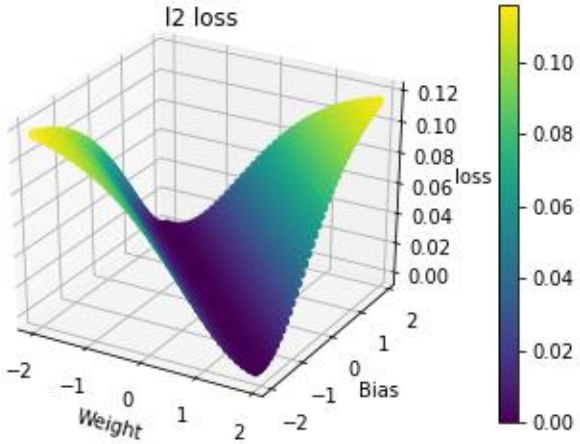

Figure 2 L2 Loss

1.3 Back-propagation with L2 loss

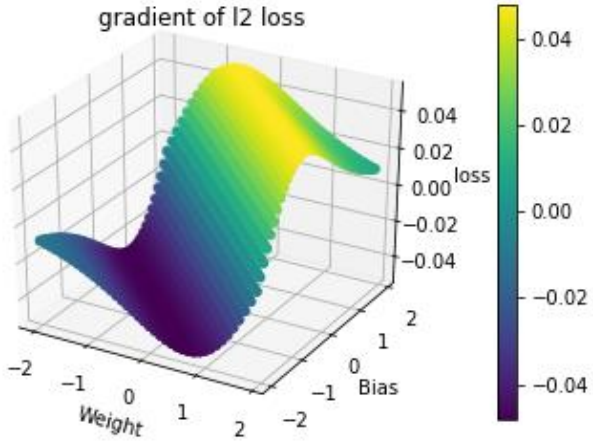

Figure 3 Back-propagation with L2 loss

1.4 Cross-entropy loss

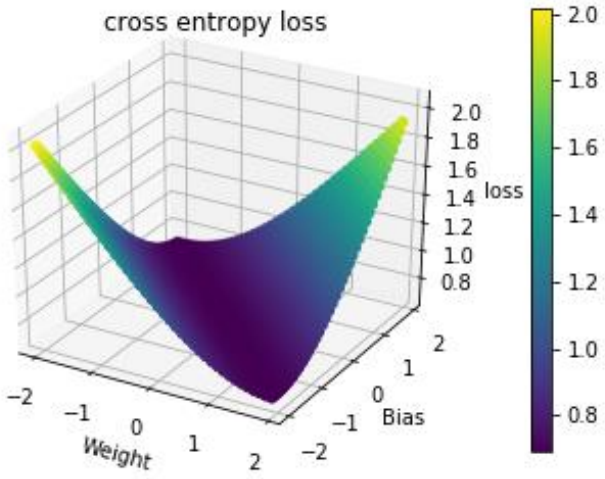

Figure 4 Cross-entropy loss

1.5 Back-propagation with cross-entropy loss

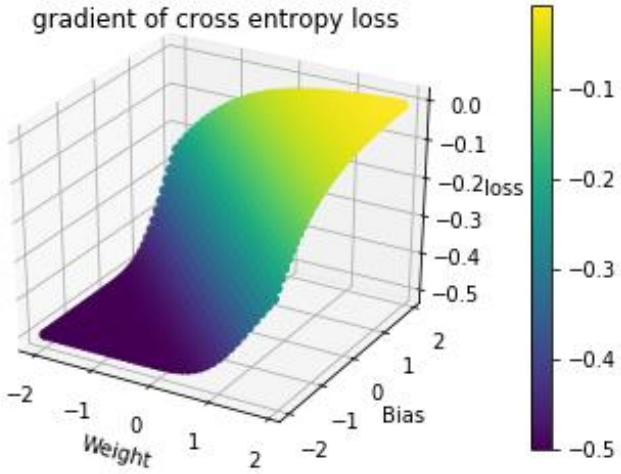

Figure 5 Back-propagation with cross-entropy loss

1.6 Explain what I observed from the above 5 plots.

- What's the difference between cross-entropy loss and L2 loss?
  The learning rate of L2 loss is slow initially. The learning rate of cross-entropy loss is much faster initially.

- What's the difference between the gradient of cross-entropy loss and the gradient of L2 loss?
  If we use Sigmoid as activation function, the gradient of L2 loss contains a sigmoid derivative in the gradient, which may result in vanishing gradient problem. It is because the gradient for Sigmoid function for points far from 0 is small. The gradient of cross-entropy loss doesn’t contain a sigmoid derivative.

- Predict how these differences will influence the efficiency of learning.
  Since L2 loss is slow initially, the speed of l2 loss for learning is much slower than cross-entropy loss for to achieve the same accuracy.

2. Experiment with Fully Connected Network (20%)

2.1 Use Sigmoid function as neuron activation and L2 loss for the network

Figure 6 Loss vs Training Iteration_2.1

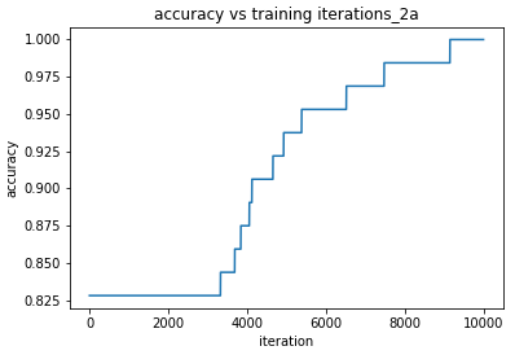

Figure 7 Accuracy vs Training Iteration_2.1

2.2 Use Sigmoid function as neuron activation and cross-entropy loss for the network.

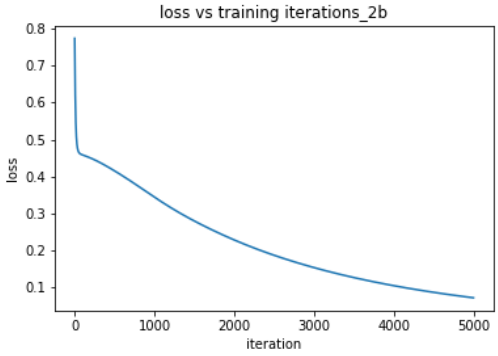

Figure 8 Loss vs Training Iteration_2.2

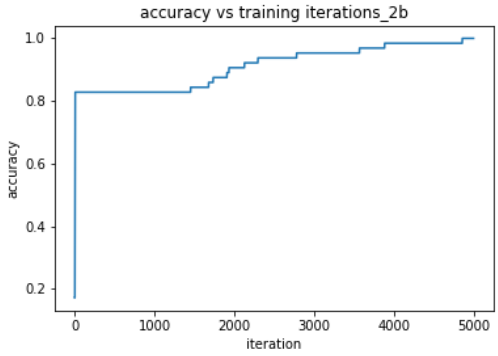

Figure 9 Accuracy vs Training Iteration_2.2

2.3 Use ReLU function as neuron activation and L2 loss for the network.

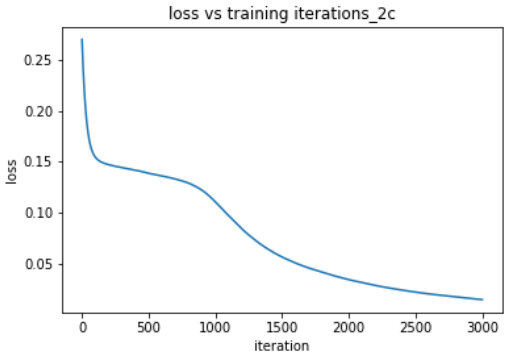

Figure 10 Loss vs Training Iteration_2.3

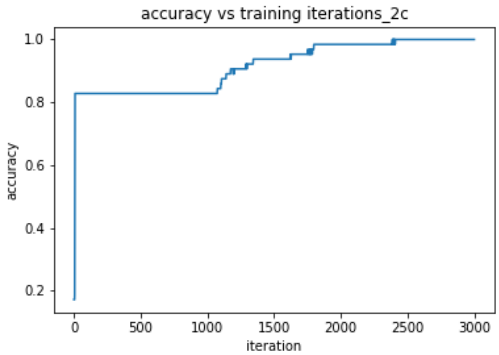

Figure 11 Accuracy vs Training Iteration_2.3

2.4 Use ReLU function as neuron activation and cross-entropy loss for the network.

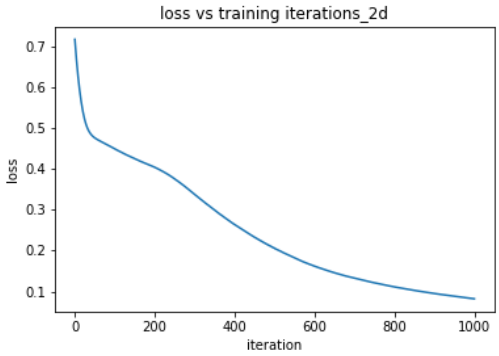

Figure 12 Loss vs Training Iteration_2.4

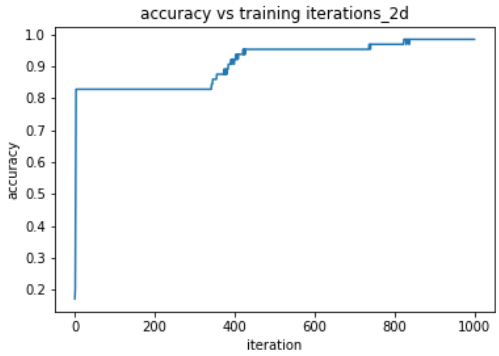

Figure 13 Accuracy vs Training Iteration_2.4

2.5 No. of iteration to reach 100% accuracy is shown in the following table:

Neuron Activation Function	Loss Function	No. of iteration	Ranking
ReLU	Cross-entropy	Approx. 800	1
ReLU	L2	Approx. 2300	2
Sigmoid	Cross-entropy	Approx. 5000	3
Sigmoid	L2	Approx. 10000	4

ReLU vs. Sigmoid: One major benefit of ReLU is the reduced likelihood of the gradient to vanish. This arises when a>0. In this regime the gradient has a constant value. In contrast, the gradient of sigmoid becomes increasingly small as the absolute value of x increases. The constant gradient of ReLU results in faster learning in my experiment.

Cross-entropy Loss vs. L2 Loss: As discussed in previous question, the learning rate of L2 loss is slow initially. The learning rate of cross-entropy loss is much faster initially.

3. Experiment with Convolutional Network (15 %)

3.1 I construct a network with two convolutional layers and one fully connected layer, concatenated with a cross-entropy loss, and used ReLU as the activation functions of convolutional layers.

When using a learning rate of 0.1, the network seems overfitting. The loss and accuracy are shown as follows.

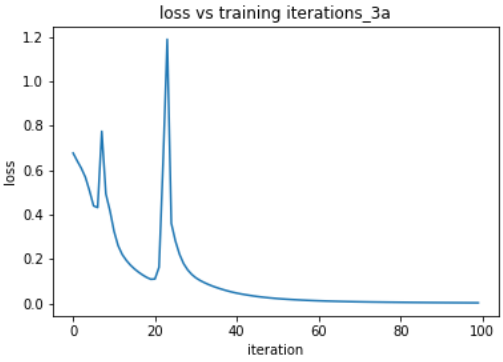

Figure 14 Loss vs Training Iteration_3.1(lr = 0.1)

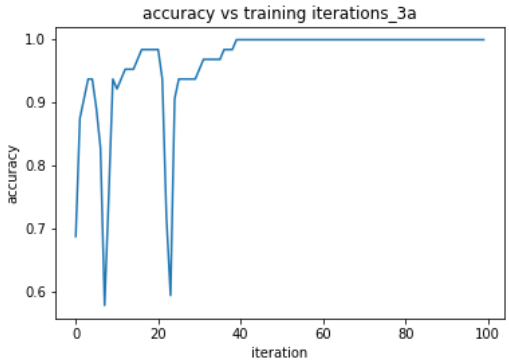

Figure 15 Accuracy vs Training Iteration_3.1(lr = 0.1)

The learning rate is set to 0.01 to prevent over-fitting. The loss and accuracy are shown as follows.

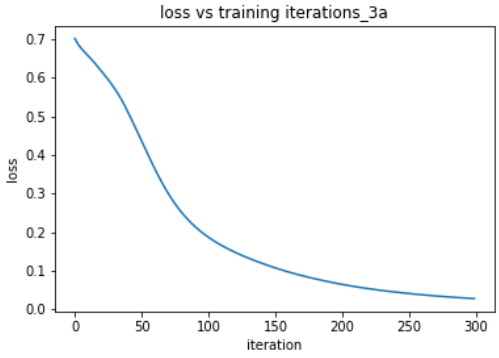

Figure 16 Loss vs Training Iteration_3.1(lr = 0.01)

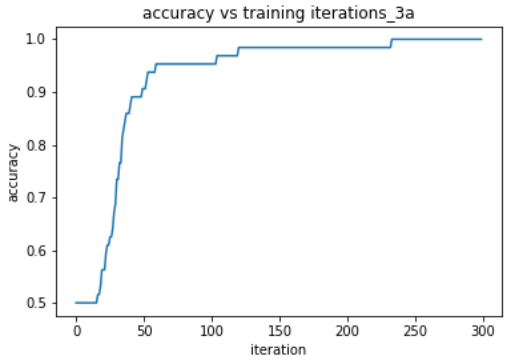

Figure 17 Accuracy vs Training Iteration_3.1(lr = 0.01)

I used a fully connected layer with a cross-entropy loss and used ReLU as the activation functions in question 2. And the convergence rate to 100% accuracy is approximately 800 iterations. While in this equation with a convolution layer, the convergence rate to 100% accuracy is approximately 250 iterations for learning rate of 0.01. Therefore, one could conclude that the convergence rate of convolutional network is faster than fully connected network. Fully connected neural networks are good enough classifiers, however they aren't good for feature extraction. Before the emergence on CNNs the state-of-the-art was to extract explicit features from images and then classify these features.

3.2 I added one more fully connected layer on top of the convolutional layer in 3.1. And the loss and accuracy are shown as follows.

After adding the fully connected layer, the learning speed up to 100% accuracy decreases. I think the decrease in speed is caused by us introduced more parameters and an additional network.

The results of using learning rate of 0.1 are shown as follows. The result is over-fitting.

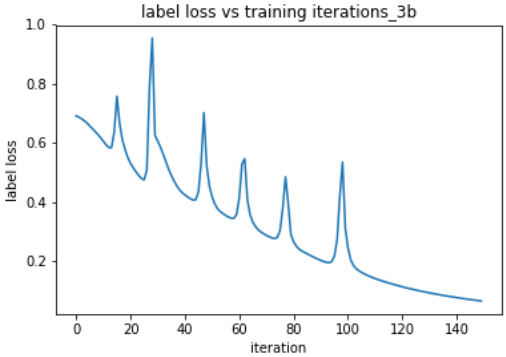

Figure 18 Cross-entropy Loss vs Training Iteration_3.2

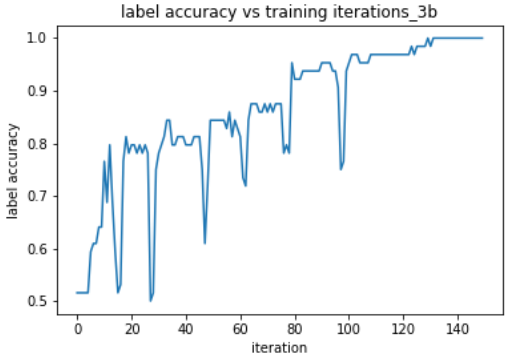

Figure 19 Classification Accuracy vs Training Iteration_3.2

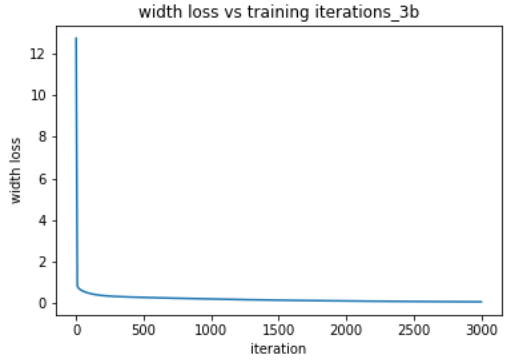

Figure 20 L2 Loss vs Training Iteration_3.2

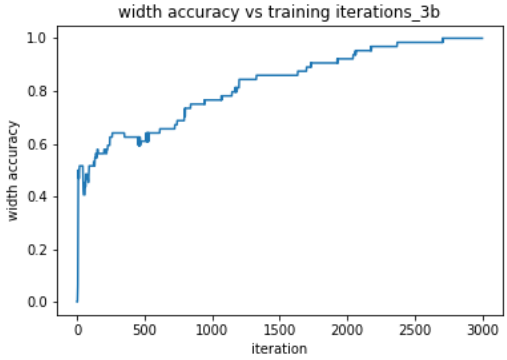

Figure 21 Regression Accuracy vs Training Iteration_3.2

The results of using learning rate of 0.01 are shown as follows. Decreasing the learning helps to prevent over-fitting.

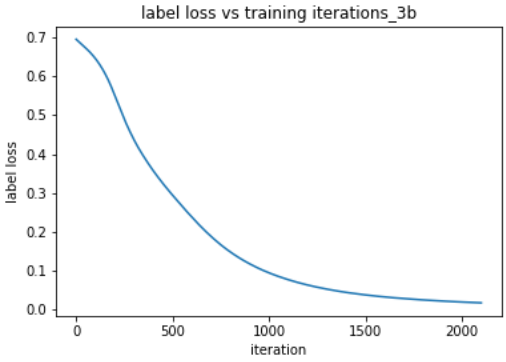

Figure 22 Cross-entropy Loss vs Training Iteration_3.2

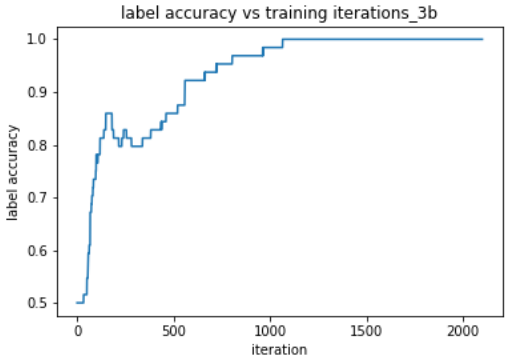

Figure 23 Classification Accuracy vs Training Iteration_3.2

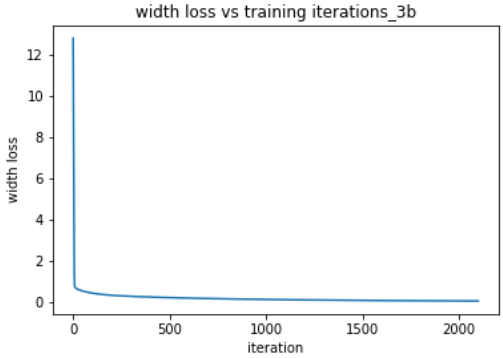

Figure 24 L2 Loss vs Training Iteration_3.2

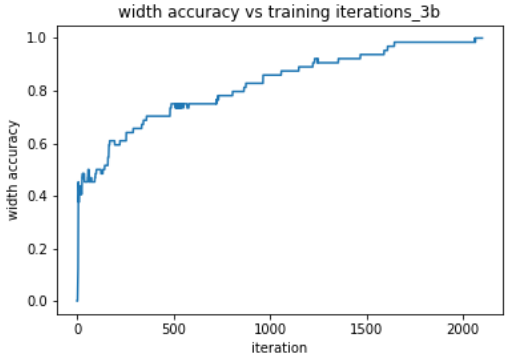

Figure 25 Regression Accuracy vs Training Iteration_3.2