The Report For Experiment One: MLP

TianYi Ma 2017011408[1] *

[1]Computer Science and Technology. Tsinghua University

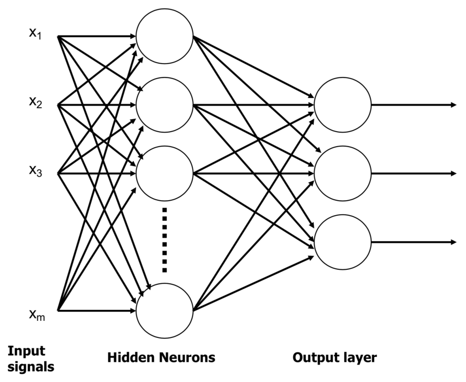

Fig. 1. Basic Architecture. It contains at most two linear layers.

In Experiment One, I construct a Multi-Layer Perception Neural Network to recognize the handwritten numbers. Most parts of this network have been constructed, what I have to do is to finish the leftover part: the layer's forward and backward functions as well as the loss functions. After finishing the Linear layer, the Relu activation layer, the Sigmoid activation layer, and the EuclideanLoss function as well as SoftmaxCrossEntropyLoss function, I conduct a series of experiments to look for the best hyper-parameters' value. After finding out the values that can conduct the best performance, several sets of experiments are designed to decide which activation function and loss function can lead to the best performance.

Construction of the Network. The network contains one or two linear layers, with each company an activation function. Mostly, the forward and the backward functions of the linear layer and the activation layer is built with numpy. After mapped by the linear and non-linear layers, the input images are turning into the result of the classifying process and the loss will be calculated. Two types of loss function is constructed. Each time, one of them will be applied to the network architecture to test which one of them can lead to better performance. In all the experiments, I set the minibatch size as 50 and the max epoch as 1000.

Adjusting Hyper-parameters. The following three experiments are aimed to target the best hyper-parameter values. In the three experiments, the network consists of one linear layer with relu activation function and SoftmaxCrossEntropyLoss function. One hyper-parameters value is changed each time with others remain the same.

Adjusting Hyper-parameters: Learning Rate. Multiple hyper-parameter should be adjusted to enhance the performance of the network. Among them, one of the most important parameters is the learning rate. To find the best setting,

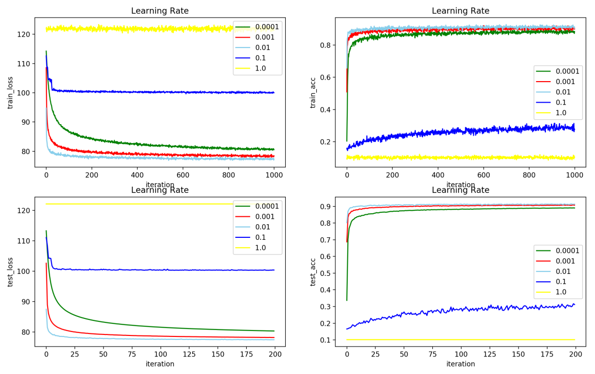

Fig. 2. The Loss and Accuracy of Different Learning Rate in 1000 Epochs

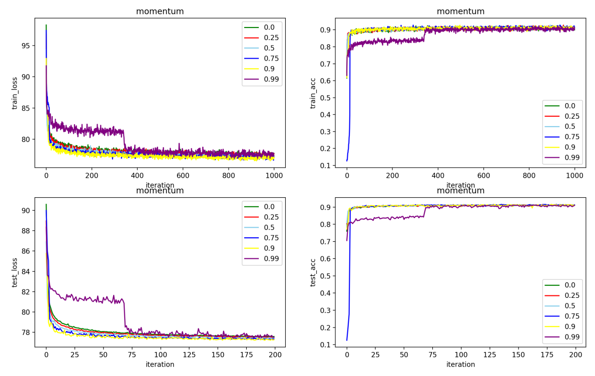

Fig. 3. The Loss and Accuracy of Different Momentum Values in 1000 Epochs.

several tests are made and the result is shown in (Fig. 2). The minibatch size is 50 and the momentum is 0.5 for all the tests and the network is trained for 1000 epochs. Based on the result depicted in the graphs, we can surmise that the rate 0.01 and 0.001 in the top 2 best choices since they are capable to produce the lowest loss value on test and training set while the highest accuracy.

Adjusting Hyper-parameters: Momentum. Momentum is another parameter that could cast a noticeable influence on the performance of the network. Keeping the learning rate 0.01 and minibatch size 50, we train the network for 1000 epochs. And the result is depicted in (Fig. 3). We can surmise from the result that 0.5 is the best momentum rate for these experiments since the network with this value can achieve the highest accuracy in both the training and testing sets.

Adjusting Hyper-parameters: Weight Decay. Weight decay is another key factor in the performance of the network.

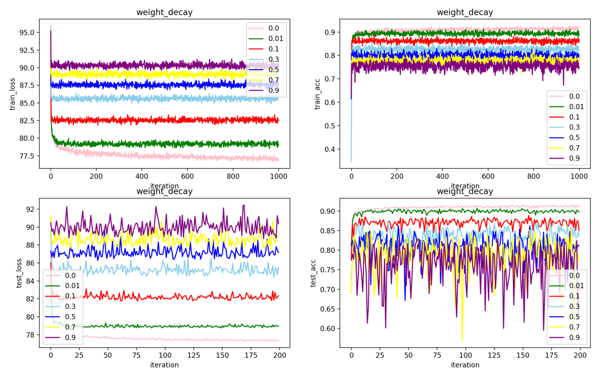
Fig. 4. The Loss and Accuracy of Different Weight Decay Values in 1000 Epochs.

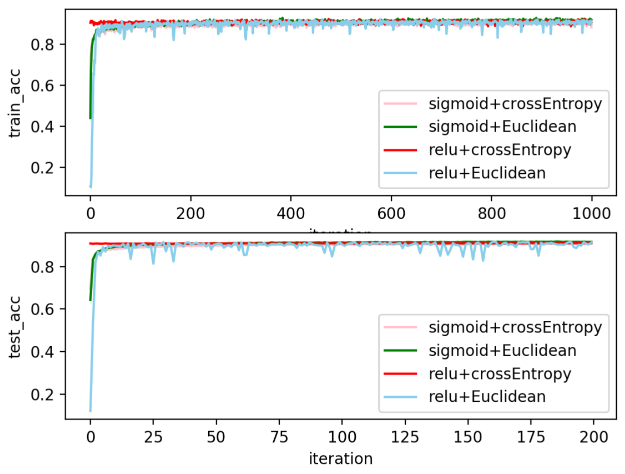
Fig. 5. The Accuracy in 1000 Epochs With Different Activation and Loss Function.

We set the learning rate of 0.01 and momentum 0.5. The result is depicted in (Fig.4). Set weight decay value 0 is the best choice since the higher the weight decay is, the lower the accuracy will be.

One Linear Layer Experiment: Compare Different Loss and Activation Functions. Based on the three experiments done above, we can determine the basic hyper-parameters' values. The learning rate is set with 0.001 and the momentum is set with 0.5 while the weight decay value remains 0. Then, four Linear layer networks are constructed. They are built with different activation functions and loss functions. The activation and loss functions pair are (sigmoid, cross-entropy), (sigmoid, Euclidean), (relu, cross-entropy), (relu, Euclidean). The accuracy values are depicted in (Fig.5). Since the loss between the two-loss functions cannot be compared, the loss values are depicted in (Fig.6) and (Fig.7). Based on the results, it can be surmised that when using the Relu activation function, the network is capable to converge faster. This is natural since the gradient of the Relu function is always 1 when the input value is greater than 0. However, when the absolute value of the input vector is greater than 1, the gradient can be very small. With a smaller gradient value, the network shall converge slower. Also, we notice that when using the same activation function, the cross-entropy loss function can enhance the converging speed of the network. The reason might be that the CrossEntropy loss can fit the classification mission better. As for the time for calculating, CrossEntropy loss function can be a bit faster than the Euclidean loss function. But the difference is fine. Observing the loss values only, we can note that the cross-entropy loss can produce a smaller loss when using the relu function. Meanwhile, the Euclidean loss compared with the sigmoid function value can bring out a smaller loss value. However, we cannot conclude that cross-entropy loss with relu and Euclidean loss with sigmoid can bring out a better result since the ultimate goal is to achieve a higher accuracy rate. We have to compare the peak value of the accuracy rate to conclude. And the result is depicted in (Fig.8). Based on the crest values, we can now conclude that cross-entropy loss function performs bet-

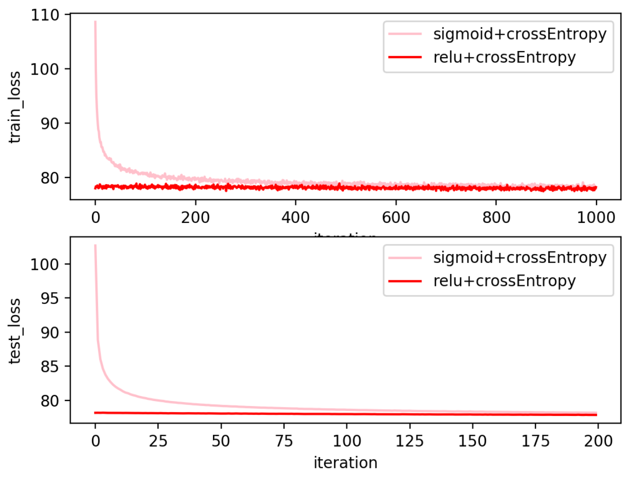
Fig. 6. The Loss 1000 Epochs with CrossEntropy Loss and Different Activation Functions.

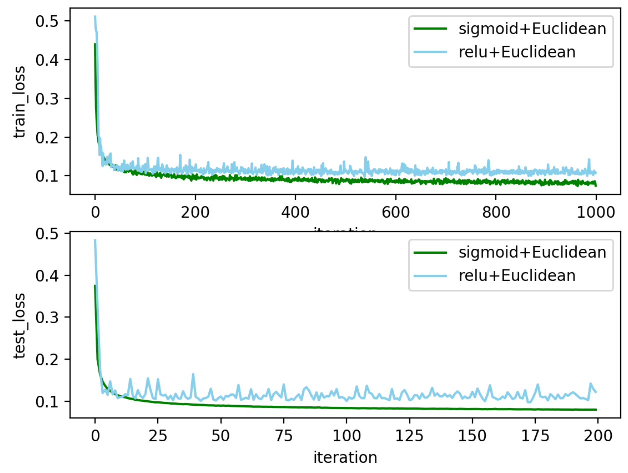
Fig. 7. The Loss 1000 Epochs with Euclidean Loss and Different Activation Functions.

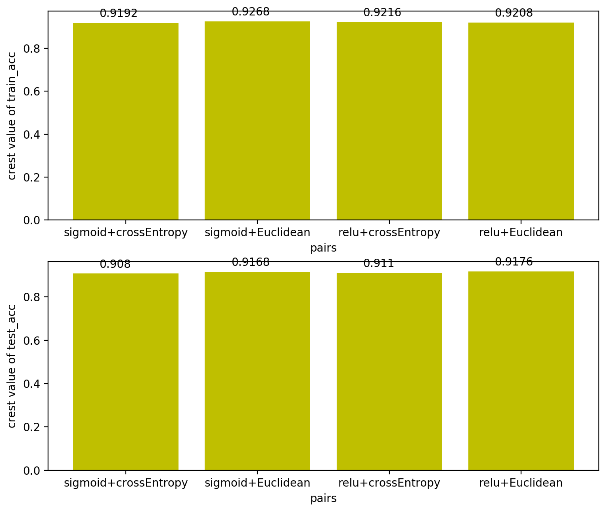

Fig. 8. The Peak Accurate Values.

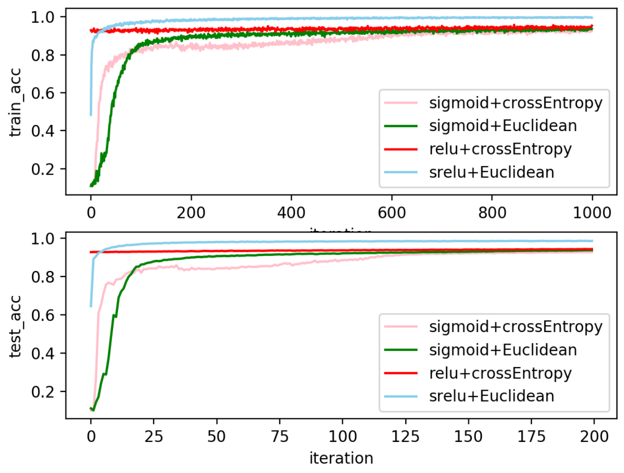

Fig. 9. The Accuracy Rates in Experiment Two.

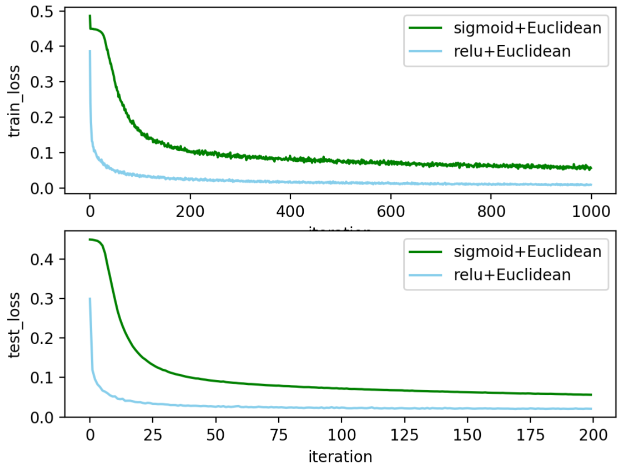

Fig. 10. The Loss Values of Euclidean Loss Function in Experiment Two.

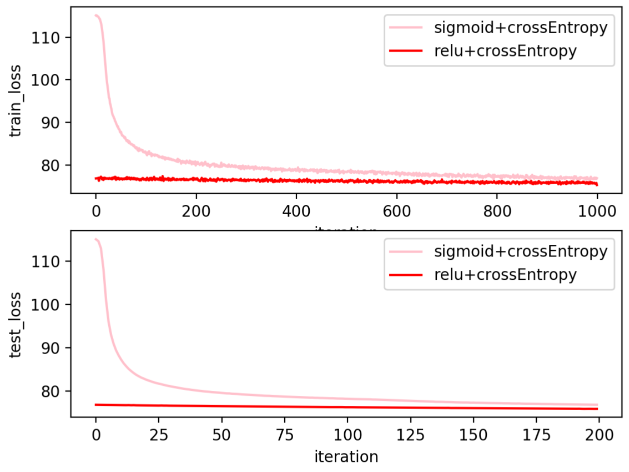

Fig. 11. The Loss Values of CrossEntropy Loss Function in Experiment Two.

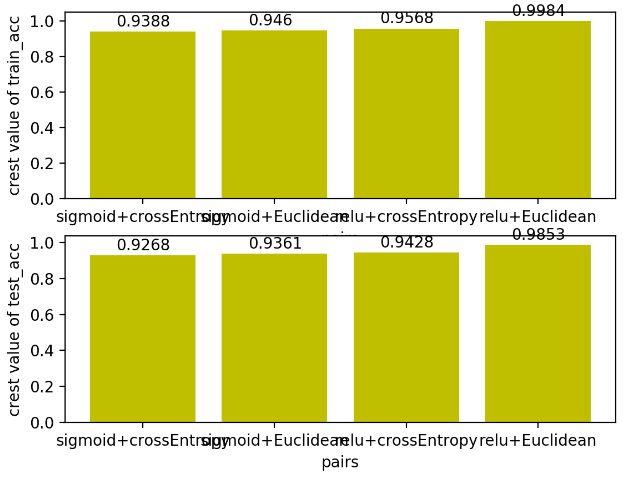

Fig. 12. The Peak Accurate Rates of Experiment Two.

ter with relu activation function and the Euclidean loss function achieves better results with sigmoid function.

Double Layers Network Experiment: Compare Different Loss and Activation Functions. In this section, we construct a 2 layer network. The first layer's input dimension is 784 and the output dimension is 256. The second layer's input dimension is 256 and the output dimension is 10. Similar to the single-layer network experiments, we try different activation functions and loss functions to find the collocation that is capable to bring out the best performance. The momentum value is 0.5, the learning rate value is 0.001 and the weight decay value is 0. The network is trained for 1000 epochs. The accuracy rates in the 1000 epochs are depicted in (Fig.9). The loss values are depicted in (Fig.10) and (Fig.11). The peak values of the accuracy rates are depicted in (Fig.12). Similar to the experiment one, we can conclude that when using the relu activation function, the network will converge faster. The reason for that phenomenon is also that the relu activation function's gradient value is greater than that of the sigmoid. Different from experiment one, both loss functions bring out better performances when using the relu activation function. When using relu activation function and Euclidean loss function, the network can achieve the best performance. Also, adding an extra linear layer and activation function can enhance the performance of the network. The reason for the phenomenon is that the two-layer network contains many more parameters and can fit a more complex function. However, since the two layers network contains more parameters, it will be more difficult for it to converge. Also, for each epoch, it will take a longer time to calculate for forward and backpropagation.

Conclusion. After constructing the networks by myself and conducting multiple experiments, I earn a better understanding of how the multi-layer network work. I also get a deeper understanding of the differences between sigmoid function and relu function as well as Euclidean loss function and CrossEntropy function.