Deep Learning for Image Analysis
Assignment 2
1MD120 61612 VT2024

Georgios Tsouderos

April 2024

Contents

1 Exercise 1. Partial derivatives

Derive expressions for

$$\frac{\partial J}{\partial b_m} \text{ and } \frac{\partial J}{\partial w_{m_j}}$$

expressed in terms of $\frac{\partial J}{\partial z_{i_m}}$, $\frac{\partial z_{i_m}}{\partial b_m}$ and $\frac{\partial z_{i_m}}{\partial w_{m_j}}$ only.

1.1 Solution

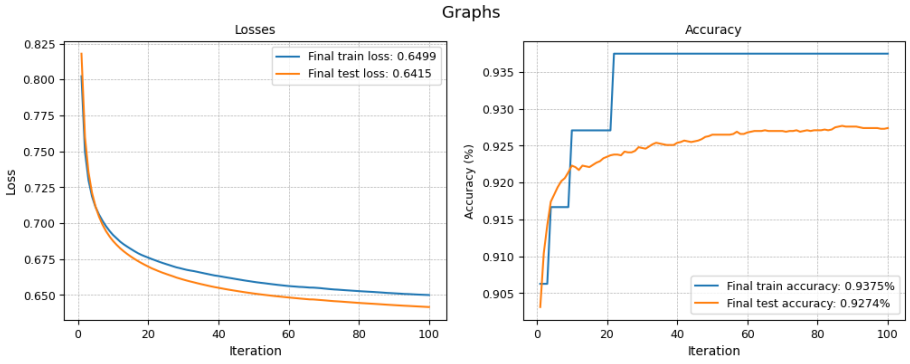
We will use the chain rule to express the above expressions in the needed terms.

$$\frac{\partial J}{\partial b_m} = \sum_{i=1}^{n} \frac{\partial J}{\partial z_{i_m}} * \frac{\partial z_{i_m}}{\partial b_m} \tag{1}$$

$$\frac{\partial J}{\partial w_{m_j}} = \sum_{i=1}^{m} \frac{\partial J}{\partial z_{i_m}} * \frac{\partial z_{i_m}}{\partial w_{m_j}} \tag{2}$$

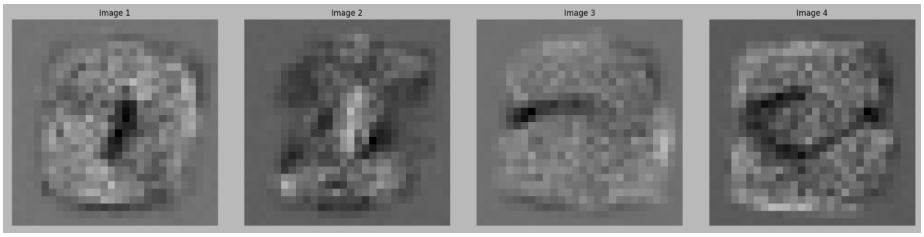
2 Evaluation of one-layer feed forward neural network

The plots below show the training and test losses and accuracy each iteration of the dataset.



Observing the above graphs, we can firstly see that the final test loss is smaller than the final train one. In addition, there is a ladder effect on the train accuracy graph. Both of these, raise some suspicions about the correct implementation of the code but the values are reasonable enough.

When reshaping the weight vector to match the dimensions of the MNIST images and plotting the result, we get:



This images basically represent the filters that are applied to the images to find the closest match. We can see that they look like numbers from 0-9. When they are applied, the image with the closest match is selected and is assigned the predicted label.

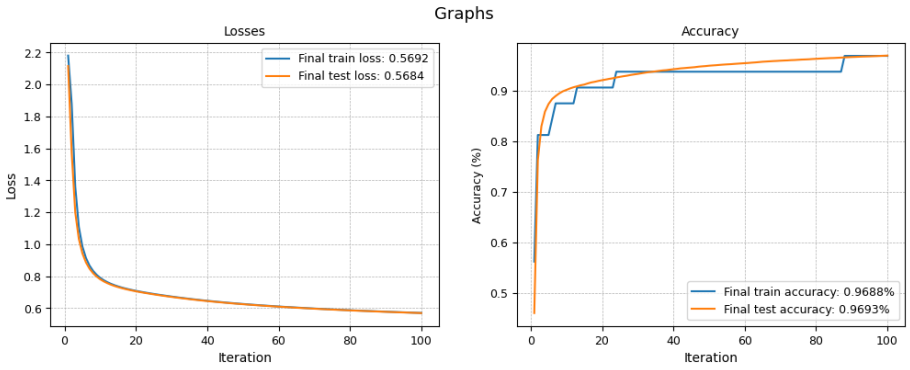
3 Multilayer Neural Network

For the implementation of a MLP we used two layers instead of one. After careful examination of the performance of the network on the validation set, the following hyper-parameters were chosen:

- Learning rate: 0.05
- batch size: 64
- epochs: 100

It was observed that the network converged faster with a higher batch size and learning rate but it reached a lower accuracy than the current model after training. The reduce of the learning rate when reducing the batch size can be justified since with less samples, the gradients can be more noisy.

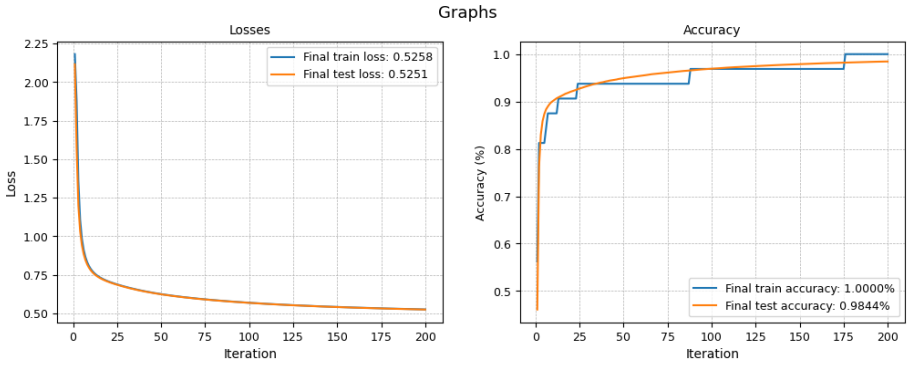
The plots below show the training and test losses and accuracy each iteration of the dataset.



Observing the above graphs, we can deduce that the model is well trained. The test set performance follows pretty closely the performance achieved from

our model with the training data. In fact, we can see that in some iterations, the accuracy achieved in the test data was higher than the corresponding one in the train data. However, the difference is quite small and can be justified due to the lower number of images in the test set.

By observing the trend of the graphs going downwards, we kept the hyperparameters the same but increased the epochs to 200 and the following performance was achieved:



We can see that the network learns the data completely, achieving accuracy of 1, even though it is probably rounded up. In addition, we achieve 0.98 accuracy on the test set.