NEURAL NETWORKS AND DEEP LEARNING

HOMEWORK 1

Yukselcan Sevil

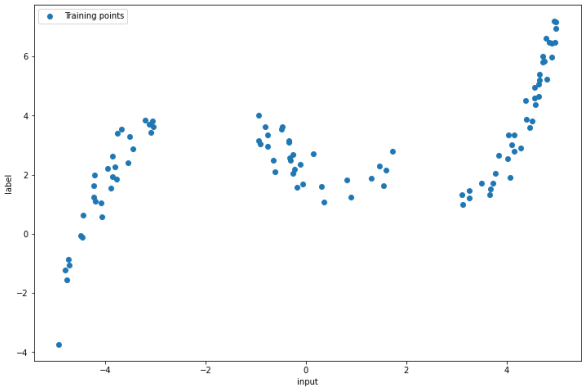
1216127

Introduction

The main goal of this homework is to implement and test the neural network models for solving supervised problems. The homework consists of two parts. The first task is a regression task that consists of a simple function approximation. The second task is a classification task that consists simple image recognition problem to correctly classify images of handwritten digits of the MNIST dataset.

1. Regression Task

The data sets given in this task consist of 100 training points and 100 test points. Training points have only noisy measures from the target function. As can be seen from the picture, the data around -2 and 2 are missing.

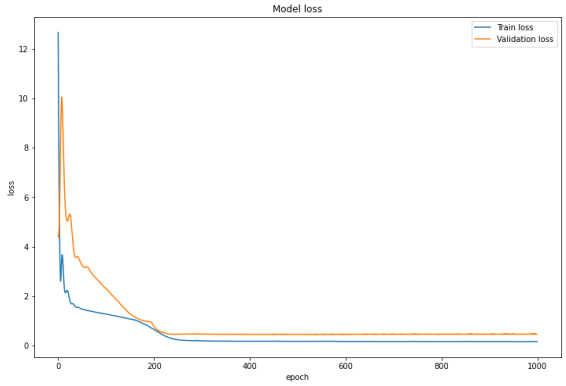


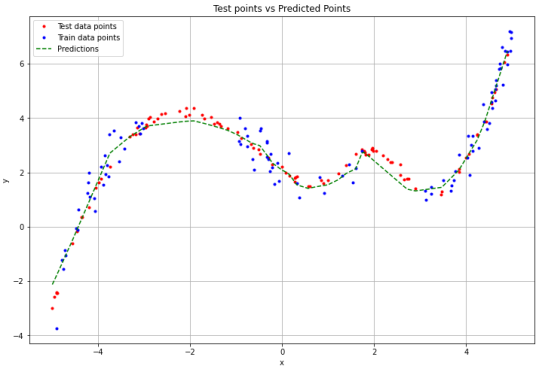
The network consists of 3 connected layers. Input and output layers just have one neuron. Because the data has real numbers and the output must be a real number. The hidden layers have 48 and 64 neurons and ReLu activation function is used for these hidden layers. Any activation function is not used in the output layer because output can be higher than 1.

We should use cross-validation to avoid overfitting due to the lack of data. In this task, we can implement GridSearchCV which is a library function that provides to loop through predefined hyperparameters and fit the model.

Skorch is a neural network library and used for cross-validation and grid search. AdamW was used as an optimizer, and Mean Squared Error (MSE) was used as the loss function. After the implementation of the grid search, the best parameters found are:

- Nh1: 48
- Nh2: 64
- Number of epochs: 1000
- Optimizer's learning rate: 0.005
- Optimizer's weight decay: 0.0001

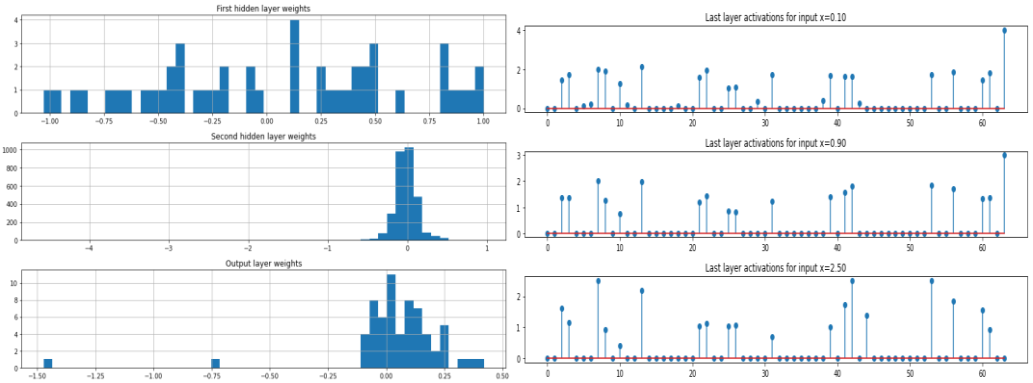




The neural network has been trained and the results are:

- Train Loss: 0.157
- Val Loss: 0.455
- Test Loss: 0.082

We can see that model can predict the values well enough around -2 and 2 after the best parameters are tuned for training. We can see that from the result, the model does not have overfitting or underfitting after training. Because train loss is quite close to test loss and validation loss is a bit higher than train loss.



For a neural network, we want to keep the values of the weights quite small. The first layer seems good because they are distributed around -1.0 and 1.0. In other layers, all the weights are distributed around 0. That means the second layer provides a prediction on the network. Also, there is no vanishing on all layers. For the layer activations, we can observe that most of the neurons are not contributed to output. Probably, the network is too complex and we can reduce the number of parameters.

2. Classification Task

The classification task consists of the classification of the MNIST handwritten dataset. The dataset consists of nine classes which are digits from zero to nine. A convolution neural network is implemented to train data. The network has three convolutional layers and max-pooling is implemented to the second convolution layer. The size of each image is 28x28x1 and the dataset has 70000 input variables. The parameters chosen for each layer are:

Conv_l1: (1, 16, kernel_size=2)

Conv_l2: (16, 32, kernel_size=2)

Conv_l3: (32, 64, kernel_size=2)

Fc1: (Conv_l3*7*7, 128)

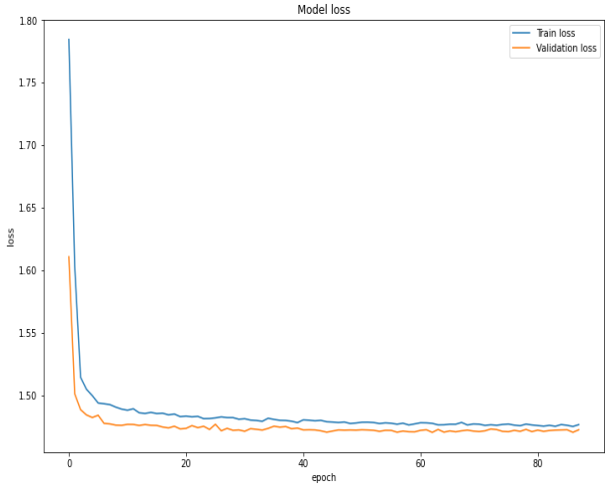
Fc2: (128, 10)

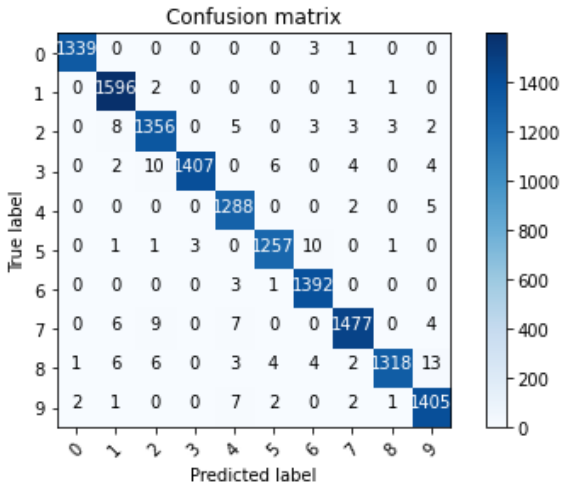
ReLU was used as an activation function for each layer except the last layer. Softmax was used at the last layer because the last layer is used for the classification of each label. Adam was used for optimization and the learning rate was set to 0.0001. Cross entropy was used as a loss function. After the optimizing hyper-parameters with GridSearchCV, the best parameters were found:

Conv_l1: 16 Number of epochs: 100

Conv_l2: 48 Optimizer__lr: 0.001

Conv_l3: 48 Optimizer__weight_decay: 0.0001

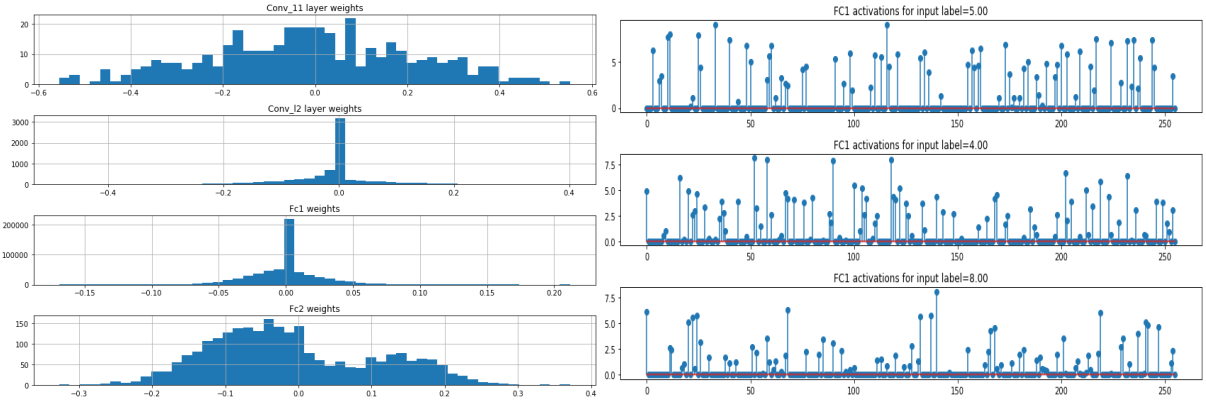




The neural network has been trained and the results are:

- Train loss: 1.477
- Validation accuracy: 0.989
- Test accuracy: 0.988

If we look at the confusion matrix, we can observe that all labels were predicted well. However, train loss is much higher than expected. Model is implemented in different ways but the result did not change. Train loss always is around at 1.5. If the train loss is large, it means that there is an under-fitting but accuracies and predictions are good.



Keeping the weights small is important in training. Weights are distributed around -0.06 to 0.06 for the first convolution layer. The weights for the conv_l2 and fc1 are much close to zero and the fc2 is around -0.3 to 0.3. That means the second layer provides a prediction on the network. For the layer activations, we can observe that most of the neurons are contributed to output. That means, most of the neurons are used by the network.

From this image, we can see the wrong predicted images with their labels. If we examine the images, we can see that most images similar to two similar digits.