Report of HW1 (Supervised Deep Learning)

Alireza Molla Ali Hosseini
(Dated: July 2022)

INTRODUCTION

In this homework I have done some tasks as mention below and I will explain them in different sections. Then at the end I will provide the results for different tasks. The codes related to this homework is provided as a Jupyter notebook (DNN-HW1).

Tasks:

- Implement basic regression and classification tasks
- Explore advanced optimizers and regularization methods
- Implement CNN for classification task
- Visualize weight histograms, activation profiles, and receptive fields

REGRESSION

First I had to download the data. The data contained 100 points with their related labels. I had to use some functions to change csv file to Dataloaders which are required inputs for pytorch networks. based on the data, the batch size and number of epoches was set to 18 and 250 respectively.

Then I implement a simple network with 3 hidden layers which first layer has 64 neurons and second one has 128 neurons (Table II). The performance of the network on training and validation data is shown in Figure 1 (a) (I split the training data to train and validation parts where the new training size was 80 percent of the previous one). The average test accuracy was **84.91 percent** (Table VI).

The next step was to tune the hyperparamets. Hence I used Optuna for this task. In order to use Optuna, one needs to define a model as trial and the parameters that wanted to be tuned (parameters were namely: number of hidden layers, number of neurons in each hidden layer, dropout rate, best optimizer method, learning rate and regularization rate). The results are gathered in Table IV

Then the best hyperparameters founded by optuna used in a deep network with mean squared loss function (since the task is regression) and the accuracy of the network was computed with K-fold Cross-Validation method (Average accuracy was **73.44 percent** and the related result is in Table VI).

The last step was to show how the network was learned in each layer by plotting the histogram of weights of different layers which is shown in Figure 1 (b),(c),and(d)

CLASSIFICATION

First I needed to download the data (which were Fashion MNIST dataset) then transfer them to tensors and prepare them for pytorch neural networks (make Dataloaders). There are 60000 pictures of 28*28 frame for training and 10000 for test with 10 different classes. The number of epoch and batch size were set to 25 and 256 respectively.

Then I defined a simple neural network to classify the data which its accuracy was **89 percent** (The parameters of the network is in Table III).

After that, I used Optuna to find the best hyperparameters for classification task. The procedure was similar to the regression part and the results are provided in Table V

Next step was to use hyperparameters of the best model and implement another network for classification with negative-log-likelihood loss function (since the task is classification) and using Cross-Validation method for finding the accuracy of the model which it was **88.45 percent**. It is also in Table VI.

Last step was to used weights of the layers of the model to show the histogram of them and

Parameters	Value
Batch size	256
Epoch	20
Convolutional layers	3
Linear layers	3
learning rate	1e-3
Weight Decay	1e-4
Drop out	0.2
kernel size	3
stride	2
padding	1

TABLE I: Parameters of CNN

Parameters	Value
Batch size	18
Epoch	250
Linear layers	3
learning rate	1e-3
Weight Decay	1e-4

TABLE II: Parameters of Basic Regression

visualize how the network was learned which is plotted in Figure 2

CLASSIFICATION WITH CNN

In this task I only needed to implement a CNN (Convolutional Neural Network) and use the classification's data and find the accuracy of this model with Cross-Validation method which was **86.82 percent** (Table VI). The parameters of this network is provided on Table I

The feature map of this network for a sample data in different layers are pictured in Figure 3

RESULTS

Parameters	Value
Batch size	256
Epoch	25
Linear layers	3
learning rate	1e-3
Weight Decay	1e-5
Drop out	0.1

TABLE III: Parameters of Basic Classification

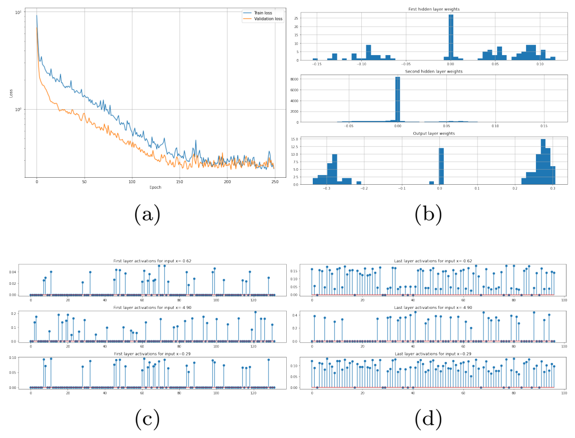
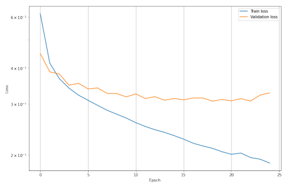
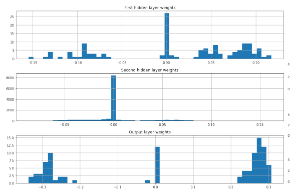


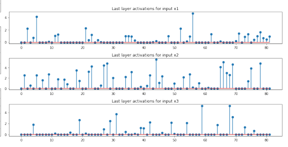
FIG. 1: (a) Regression Loss (b) Histogram of weights of regression network (c) Activation of first layer of regression network (d) Activation of last layer of regression network



(a)

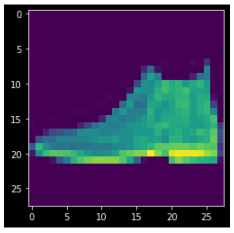


(b)

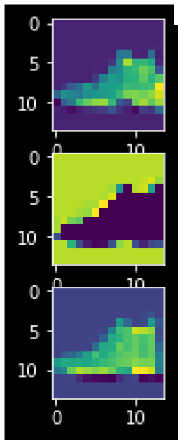


(c)

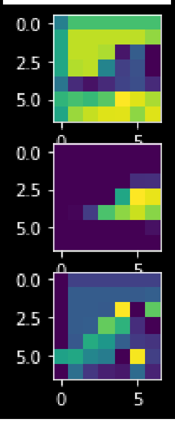
FIG. 2: (a) Classification Loss (b) Histogram of weights of classification network (b) Activation of last layer of classification network



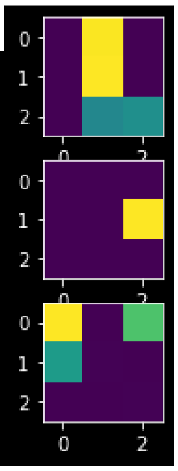
(a)



(b)



(c)



(d)

FIG. 3: (a) Sample Data for CNN (b) feature map of CNN (first layer) (c) feature map of CNN (second layer) (d) feature map of CNN (last layer)

Parameter	Range	Best
Number of layers	1-3	3
Learning rate	1e-5 - 1e-1	0.0015
Weight Decay	1e-5 - 1e-1	0.0189
Optimizer	**[Adam, RMSprop, SGD]**	**Adam**
Number of hidden units(first layer)	64-256	132
Drop out(first layer)	0.2-0.5	0.47
Number of hidden units(second layer)	64-256	99
Drop out(second layer)	0.2-0.5	0.22
Number of hidden units(third layer)	64-256	97
Drop out(third layer)	0.2-0.5	0.23

TABLE IV: Optuna search result for Regression network

Parameter	Range	Best
Number of layers	1-3	2
Learning rate	1e-5 - 1e-1	0.0022
Weight Decay	1e-5 - 1e-1	9.47e-5
Optimizer	**[Adam, RMSprop, SGD]**	**RMSprop**
Number of hidden units(first layer)	64-256	145
Drop out(first layer)	0.2-0.5	0.36
Number of hidden units(second layer)	64-256	82
Drop out(second layer)	0.2-0.5	0.2

TABLE V: Optuna search results for Classification network

Network	Accuracy
Basic Regression	84.91
Regression with 10 Fold Cross-Validation (Optuna)	73.44
Basic Classification	89
Classification with 10 Fold Cross-Validation (Optuna)	88.45
CNN with 10 Fold Cross-Validation	86.82

TABLE VI: Networks Results