Submitted by: Lior Fuks Matriculation-Nr. 4251285

Deep Learning Lab Course Exercise 1

Implementation of a feed-forward Neural Network

Introduction

This exercise task is to implement a feed-forward neural network in python. The neural network is used as a classification tool for the MNIST database. The MNIST database contains a set of images of hand written digit. This database is composed of a training set of 60000 images and a testing set of 10000 images. This exercise aim is to train a neural network and achieve the lowest classification error. This goal is achieved by tuning the neural network hyper-parameters as learning rate, epochs, batch size and more.

Results

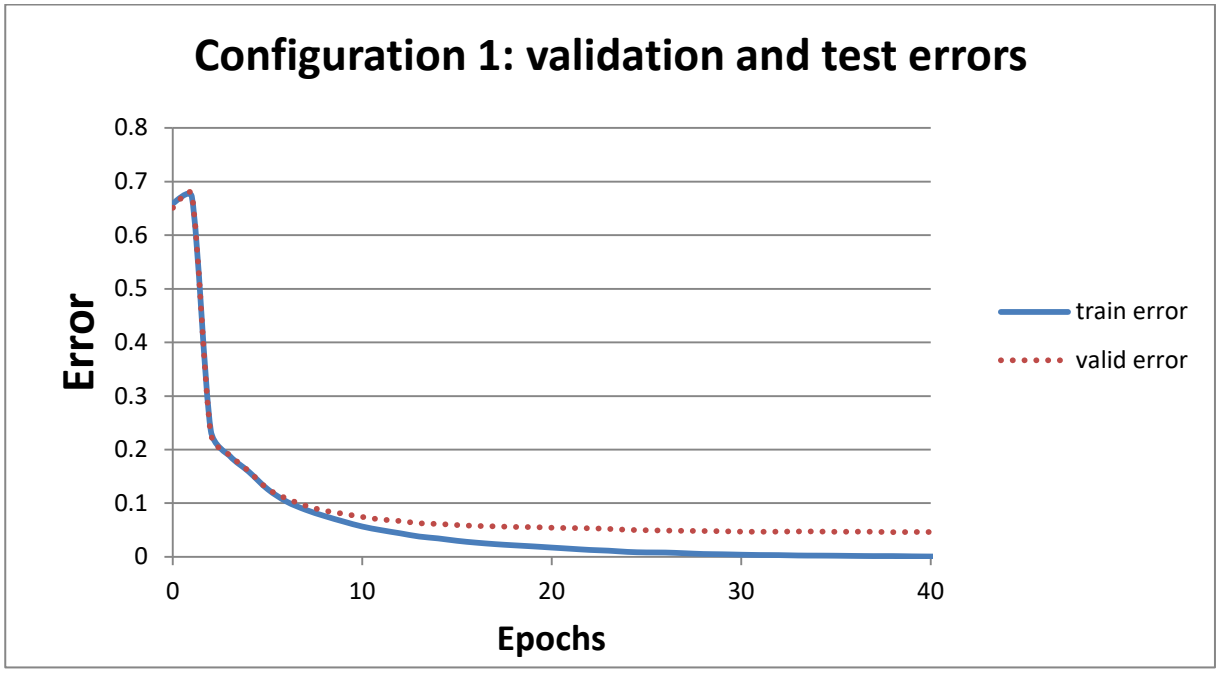
Validation and classification error

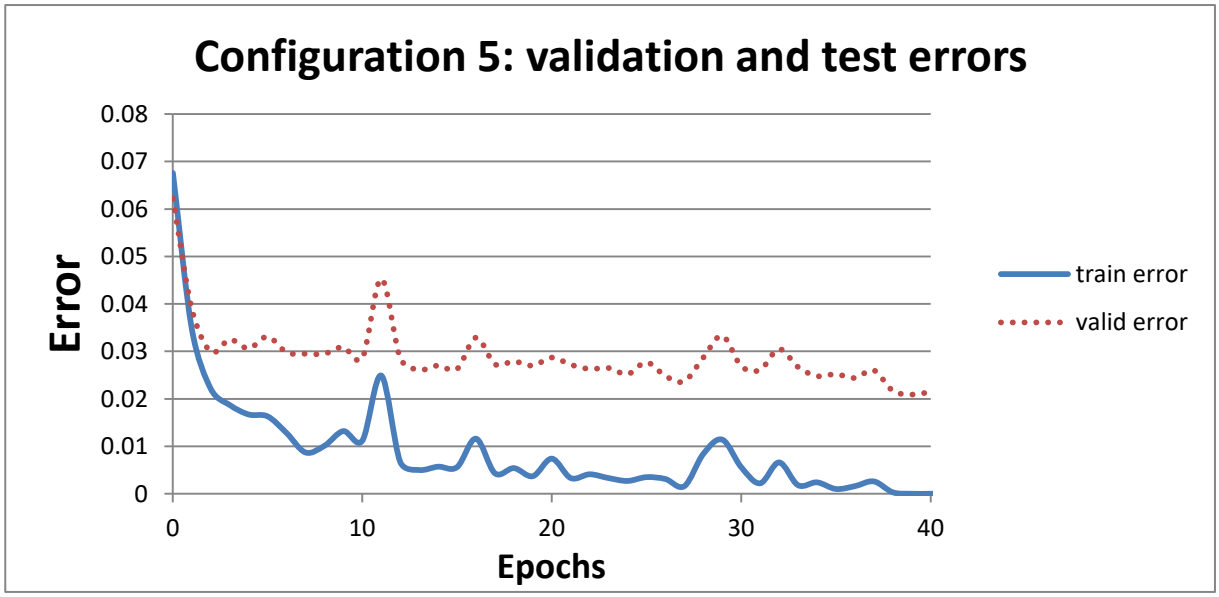
To achieve the lowest training and validation error, different parameters configurations were used. In this exercise, I decided to try to optimize the parameters: training size, learning rate and batch size. In addition to these, many other parameters could influence the classification error: activation function, number of units, number of layers and more.
In the following table, we can observe the effect on the classification error by changing the parameters: training size, learning rate and batch size.

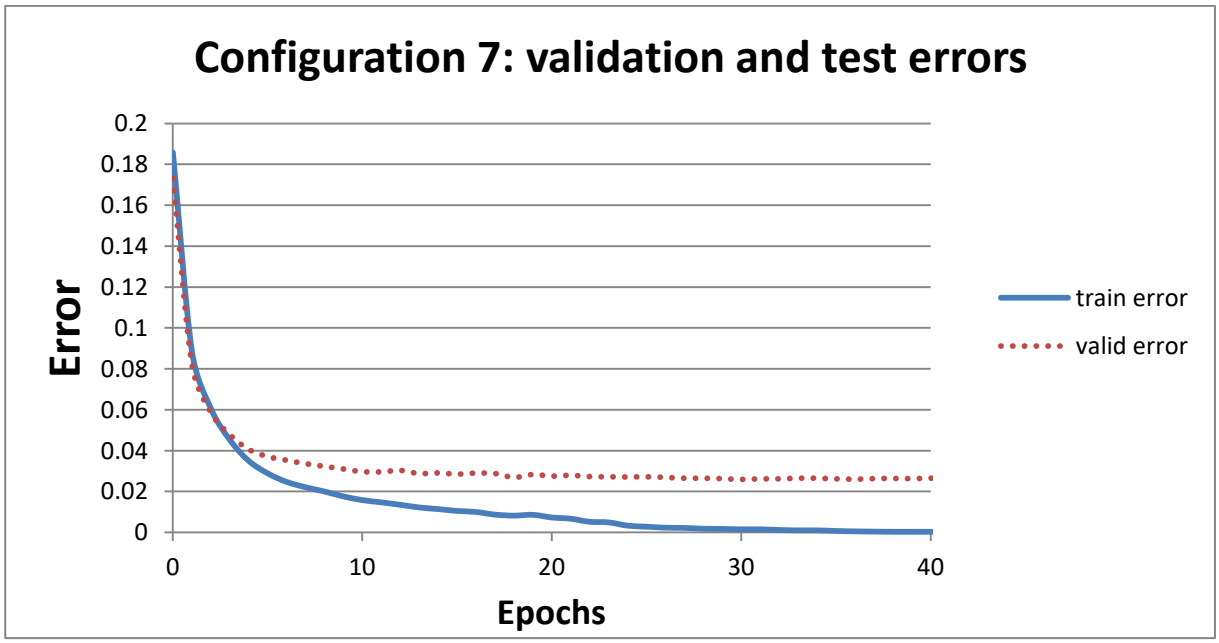
configuration	Training size	Batch size	Learning rate	Train error [%]	Valid error [%]	Test error [%]
1	10000	64	0.1	0	4.5	4.7
2	10000	64	0.01	13.4	13	14.3
3	10000	64	0.3	0	4.09	4.13
4	10000	10	0.1	0.0	3.51	3.57
5	40000	10	0.1	0.0	2.15	2.03
6	40000	10	0.3	0.9	0.9	0.9
7	40000	64	0.1	0.01	2.65	2.69

Table 1: Classification error using different hyper parameters

In the following graphs, the validation and training error is plotted for different hyper parameters configuration.


Configuration 1: validation and test errors
Error
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
10
20
30
40
Epochs
train error
valid error


Configuration 5: validation and test errors
Error
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0
0
10
20
30
40
Epochs
train error
valid error


Configuration 7: validation and test errors
Error
0.2
0.18
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
0
10
20
30
40
Epochs
train error
valid error

Classified and misclassified examples:

In the following table an example for incorrectly classified digits:

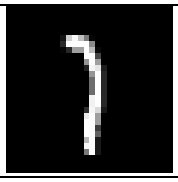
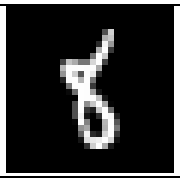
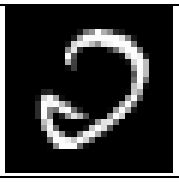
		
7 was classified as 1	8 classified as 6	2 classified as 0

Table 2: incorrectly classified hand written digits

In the following table an example for correctly classified digits:

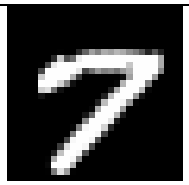
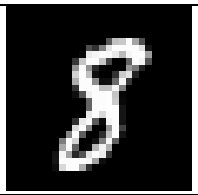
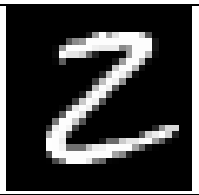
		
7 was classified as 7	8 classified as 8	2 classified as 2

Table 3: correctly classified hand written digits

Discussion:

The best testing error was achieved by the configuration: test set size – 4000, learning rate 0.1 batch size 10.

Trying different parameters yields different error results, by trying different configurations we observe that:

- Training set - Increasing the testing set from 10000 to 40000 improves the testing, validation and test classification errors.
- Learning rate – large learning rate might not converge to a local minimum where a small learning rate might take a long time to converge.
- Batch size – small batch size could inject noise to the stochastic gradient decent.

In this exercise I have searched for the optimal parameters by a trial an error, changing different hyper parameters and observing the effect on the classification error. This could be optimized by using a hyper-parameters optimization method as

- Random search – changing the parameters randomly.
- Local search – Modifying one parameter and observing the effect on the error. If the error decreased, continue to change the parameter in the same direction. If increased change parameter in the other direction
- Bayesian Optimization - By fitting a probabilistic model, the network could evaluate which parameters to explore next.