NNDL: Homework 1
Supevised Deep Learning

Alessandro Lovo

January 6, 2021

1 Introduction

This homework consists in applying supervised deep learning to two tasks: a regression task consisting in approximating a scalar function of a scalar variable and a classification task that is recognizing the handwritten digits of the MNIST dataset. For the regression task a fully connected network (FCN) will be used, while for the classification task both a fully connected but most importantly a convolutional network (CNN) will be tested. In both cases different architectures, optimization and visualization techniques and hyperparameter search will be tried.

1.1 General framework

Both tasks rely on a framework of python classes that unfolds as follows:

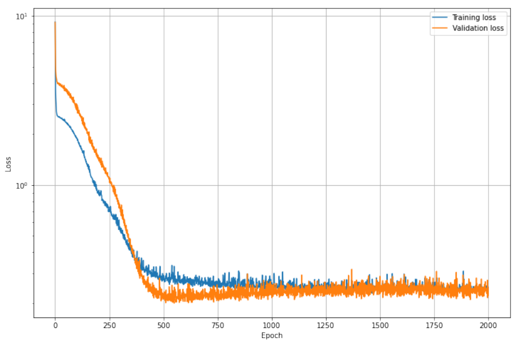
- **Net**: class inheriting from *torch.nn.Module* that contains the actual neural network with a specific architecture.
- **Evolver**: class for handling the training and validation of a *Net*. In this class there is a check at the end of every training epoch to interrupt the learning process. To implement early stopping one just needs to inherit from the *Evolver* class and specify that check condition. In particular the learning process stops if the validation loss isn't decreasing after *patience* number of epochs.
- **KFoldCrossValidator**: class for performing k fold cross validation on a particular set of hyperparameters.

2 Regression task

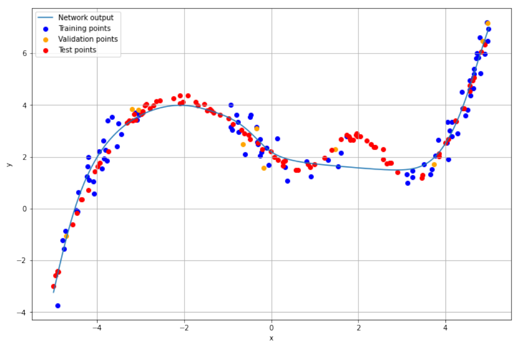
2.1 Basic solution

The data for this task consists in 100 points for training, arranged in such a way to leave two 'gaps' (), and 100 test points that fill the gaps, allowing to test how good the net is in generalizing its learning.

As a basic solution I implemented a two layered FCN with 128 neurons per hidden layer, the sigmoid as activation function and trained it for 1000 epochs using the Adam optimizer with learning rate set to 10^{-3} and a batch size of 10 datapoints. The loss function used is the mean square error (MSE) and 10% of training data is used for validation. In fig 1 are shown the results.



(a)



(b)

Figure 1: Results of the basic solution: training and validation loss as a function of epoch number (a) and Output of the trained network compared with training, validation and test datapoints (b).