# CNN with MNIST

Juliane Weilbach

13 November 2018

## 1 Introduction

This exercise is about a classification using a convolutional network (CNN) with a handwritten digit dataset, called MNIST. The implementation was done with a python template of the University of Freiburg.
The CNN consists of two convolutional layers, each followed by ReLU activation function and a max pooling layers. After the convolution layers the net has a fully connected layer with 128 units and a softmax layer. The net is optimized by a cross-entropy loss with stochastic gradient descent.

## 2 Results

### 2.1 Different Learning Rates

First of all the net was trained with different learning rates (0.0001, 0.001, 0.01, 0.1). In the first step a very small learning rate of 0.0001 was used, which includes very small steps in the Gradient Descent Approach. Twelve epochs would not be enough to get good results with this learning rate. In contrast of that a learning rate of 0.1 would mean, that the Gradient Descent converges very fast. This could be seen in figure 1. I would choose the model with a learning rate of 0.1 for this example.

### 2.2 Different Filter Sizes

In the second step the CNN was trained with different filter sizes in the convolutional layers [1]. In contrast to filter size 1 and 3 the filter size 5 and 7 lead to equally good results. The bigger the kernel-size the bigger the receptive field. This means the net is also looking at wider surrounding areas, which would lead to better results.

[1] filter sizes [1, 3, 5, 7], learning-rate set to 0.1

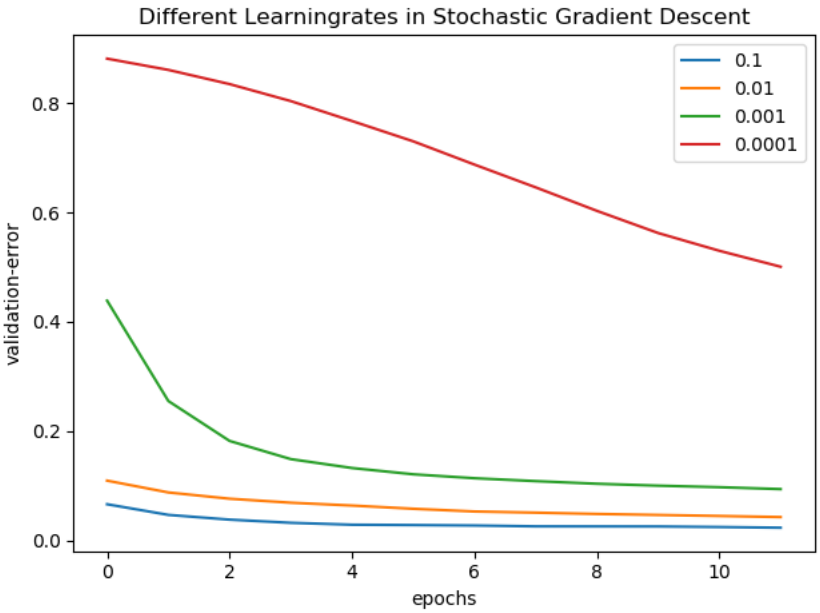


Figure 1: Validation vs. Training Loss

## 2.3 Random Search

Finally to optimize the hyperparamters (learning rate, batch size and filter size) of the CNN the random search algorithm was applied. HpBandster is used for implementation.

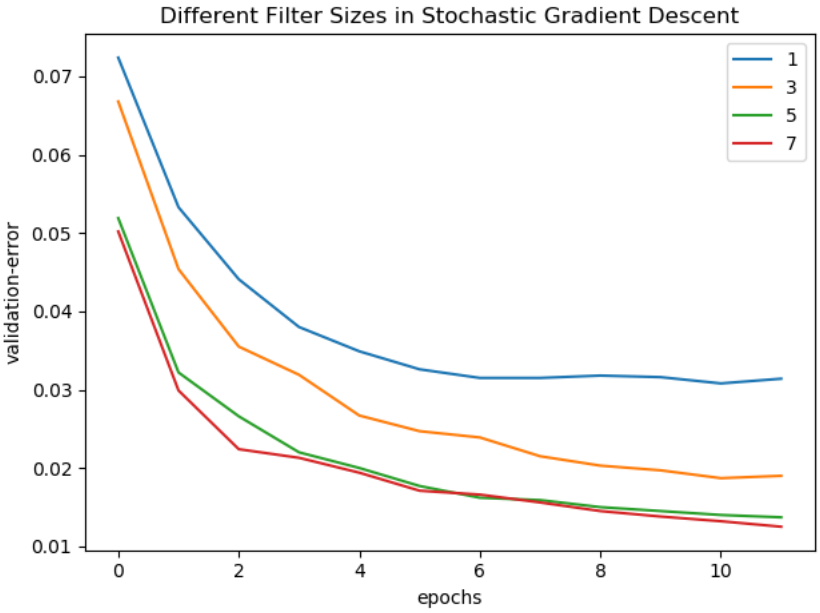


Figure 2: Validation error for different Filter Sizes