# MNIST

Homework assignments are mandatory. In order to be granted with 8 ECTS and your grade, you must pass 9 out of 10 homeworks. Upload your solution as iPython notebook (*.ipynb*) and as HTML export until Saturday 11th November 23:59 into the public *Homework Submissions* folder on studip. Name your file as *<your_group_id >*_mnist*<.ipynb/.html >*.

Further, you have to correct another group's homework until Monday 13th November 23:59. Please follow the instructions in the *rating guidelines* on how to do your rating.

If you encounter problems, please do not hesitate to send your question or concern to lbraun@uos.de. Please do not forget to include [TF] in the subject line.

## 1 Introduction

In this assignment we are going to implement a single-layer feed-forward neural network in TensorFlow, which is capable of categorizing handwritten digits.

## 2 Download

Please download the four *.gz* archive files (*train-images-idx3-ubyte.gz*, *train-labels-idx1-ubyte.gz*, *t10k-images-idx3-ubyte.gz*, *t10k-labels-idx1-ubyte.gz*) from the MNIST dataset homepage and extract them to your working directory. They contain the training dataset, the test dataset and the respective labels.

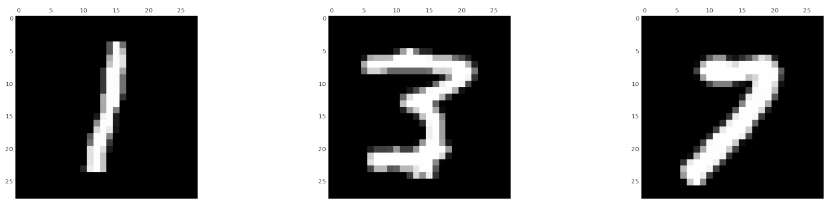
## 3 Read the data

The images are provided in a non-standardized binary format. You can download a python script from studip (*mnist-helper.py*), which reads the data for you, or you can implement your own script, following the description of the file format on the MNIST dataset homepage.

Further, the provided script separates the training data into a seperated training and validation data set and provides methods to retrieve random batches. You are free to use them or to implement your own.

# 4 Investigate the data

Use matplotlib to plot some of the handwritten digits. Do the labels fit to the images? Could we use our trained network for the categorization of handwritten digits of German students (No need to write your answer down)?



# 5 Implement the DFG

Use TensorFlow to implement the data flow graph of a fully connected single-layer neural network, with 784 inputs and 10 output neurons. Do not forget to add a bias to your network. Which of the parts should be implemented as a *tf.Variabl* and which parts should be implemented as *tf.placeholder*. Inititialize your biases with 0 and your weights from a normal distribution with a standard-deviation of 0.000002.

$$\bar{y} = \sigma(\bar{x}W + \bar{b})$$

Use the *softmax* function as the activation function for your neurons. Find the built in function in the documentation, which applies the softmax function to the drive (a.k.a logits) and calculates the cross entropy for you.

# 6 Train the network

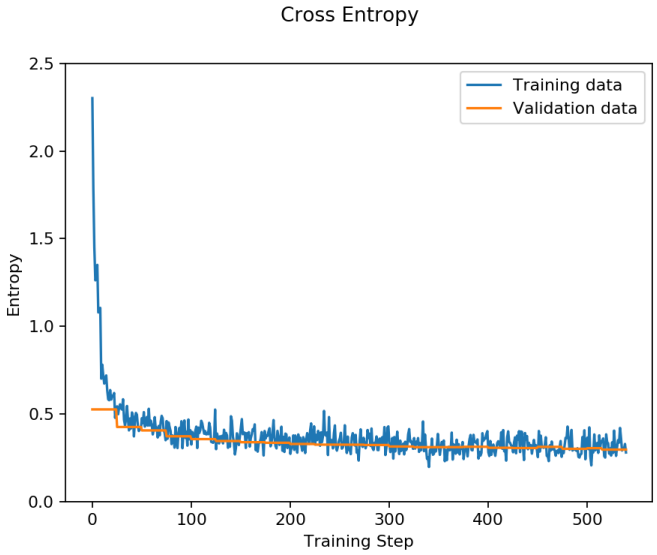
Use TensorFlow's capability of auto-differentiation to minimize the *cross entropy* of your network. Train your network with **random mini-batches**

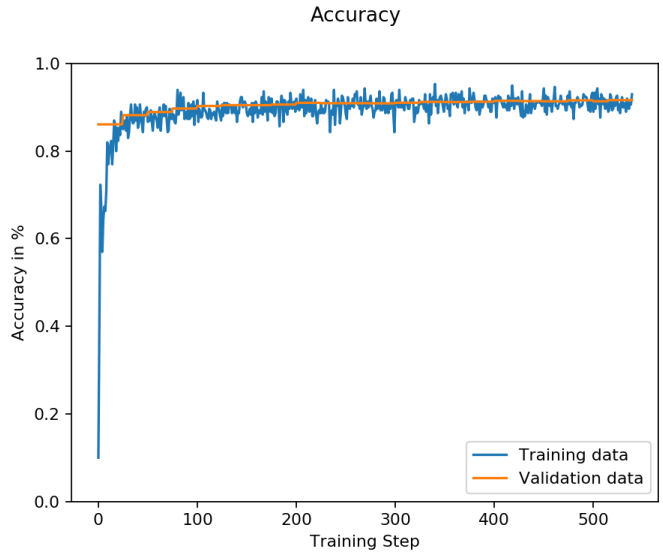
from the training data and validate the cross entropy and accuracy (Which fraction of the sample was categorized correctly?) of your network, every $n$ training steps with the validation data. Do not train your network on the validation data but only perform the forward step!

Train the network for three *epochs*. Where one *epoch* means, that the network did see every training sample once.

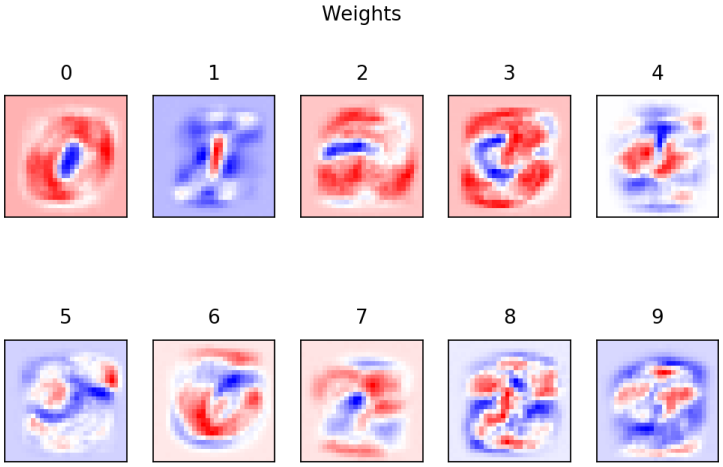
## 6.1 Monitor the training process

Plot the cross entropy and the accuracy of your network on the training and validation data against the training steps while the network is trained in order to monitor the training process.






Further, we would like to understand what our network is actually learning. Hence, we are interested in how our weights actually look like. Retrieve the weights of your network every $n$ training steps and plot them. Use a diverging colormap to be able to differentiate positive from negative weights easily. How do the weights change during the training process and which information can we retrieve from the plot (No need to write your answer down)?



# 7 Optimize the ANN

Try to modify the learning rate and the mini-batch size such that the performance of your network on the validation dataset increases. Are they related? You can also try to use different gradient descent algorithms. Once, you are unable to further improve the performance, run your model with the test dataset to estimate the application phase performance of your model.

# 8 Find help

If you struggle, there is also a tutorial on the TensorFlow homepage, which implements pretty much the same neural network. However, try to implement the neural network based on your current knowledge, the information provided on the slides and the TensorFlow python API documentation first.