Deep Learning – CNN

Summer 2023
Prof. Dr. Jochen Schmidt

Technische Hochschule Rosenheim

1) CNNs with PyTorch/MNIST

We will use the MNIST database (http://yann.lecun.com/exdb/mnist/) in this exercise. It contains (normalized) hand-written digits in images of size 28x28 as well as corresponding labels. The images are separated in training set (60,000) and test set (10,000). PyTorch contains a function for loading the data set.
A Python program template for training a simple CNN is provided for download in the Learning Campus. You can either run it using a local installation of Python/PyTorch, or run a Python-Jupyter-Notebook (e.g., on Google Colab). GPU support is not required, CPU will be sufficient.

a) The initial network structure is: conv1 - ReLU – Pool/4 - Flatten - output layer – Softmax.
   Modify the structure to:
   conv1 - ReLU - conv2 - ReLU – Pool/2 - conv3 - ReLU – Pool/2 - Flatten - fully connected - ReLU - output - Softmax

   The convolutional layers 2 and 3 should use filter sizes of 3x3 with appropriate padding and stride. The pooling layers should use 2x2 Max-Pooling.
   Also, you'll need to add an additional fully connected layer before the output layer. This requires changing the input size of the current output layer appropriately.

b) Add a dropout layer in between linear layer 1 and 2. What does a dropout do?
c) Replace the deterministic split into training and validation set to a random split
d) Randomization is disabled for easier testing. Enable randomization, run the training multiple times, observe the changes.

2) CNNs with PyTorch/CIFAR-10

This is an additional exercise, with longer training time (in particular if no GPU is used).
Change your network to train and classify on the CIFAR-10 database (http://www.cs.toronto.edu/~kriz/cifar.html). It contains 60.000 color images of size 32x32 in 10 classes (50.000 training and 10.000 test).
This is how you load the data set: https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html