DATA.ML.200 Pattern Recognition and Machine Learning

Exercise Set 3: Convolutional Neural Network (CNN)

1. `pen&paper` Count the form and number of parameters in different layers (2 points)

   a) Last week we defined a full-connected ("vanilla") neural network for $64 \times 64 \times 3$ size RGB images. The images represented traffic signs from two different classes.

   Let's now define an alternative convolutional structure:

   - The first layer is 2D convolution layer of 10 filters of the size 3×3 with stride 2 and ReLU activation function.
   - The first layer is followed by a 2×2 max pooling layer.
   - The max pooling layer is followed by another convolutional layer with the same parameters as the first.
   - The second convolutional layer is followed by another max pooling layer of the same parameters.
   - The second max pooling layer is "Flattened" and followed by a full-connected (dense) layer of two neurons with sigmoid activation function.

   Compute the output size of each layer. Also compute the total number of parameters (weights).

2. `python` *Define the network in Keras.*

   Define the homework convolutional network in your code.

3. `python` *Compile and train the net.* (8 pts)

   Use the same two class German Traffic Sign Recognition Benchmark (GTSRB) dataset from the previous exercise and the same train-test split.

   Use the following parameters:

   - **Loss:** binary crossentropy (same thing as log loss; see previous exercises)
   - **Optimizer:** stochastic gradient descent (SGD)
   - **Minibatch size:** 32
   - **Number of epochs:** 20

Compute the test set accuracy for your network. You may give the test data as validation data so that you will see the accuracy after each epoch.

Note that by calling the `.summary()` function of the compiled model you can check that your calculations of layer outputs and total number of parameters is the same.