Exercise 2 – 15st of October 2025

First Group Task

SVM, MLP and CNN

Deadline: 5th of November 2025 (end of day)

With this exercise we want you to build the foundation for your Pattern Recognition framework. To do this you will keep working on the MNIST dataset, with which you should be familiar by now. In this exercise your aim is to improve the recognition rate on the MNIST dataset using Support Vector Machines (SVM), Multilayer Perceptron (MLP) and Convolutional Neural Networks (CNN).

Reminder: You are free to use any library you like as it is more focused on optimising any method for the dataset at hand.

Important: In addition to these group exercises, there is a short individual task that you have to hand in through ILIAS. You have to calculate the forward pass of the CNN by hand. The instructions for this task are in `Exercises/Individual Exercises/2-CNN/Exercise-2-CNN-individual.pdf`.

Dataset: Full MNIST

In the previous exercises, you have used a subset of MNIST with 1 000 training and test samples. You will now move onto the full MNIST dataset, which consists of a training set with 60 000 images and a test set with 10 000 images. It is available on ILIAS in `Exercises/Group Exercises/2-SVM-MLP-CNN`.

The dataset is now given as individual images rather than a single CSV file. They are split into two directories, `train` and `test` and in each directory they are grouped by their classes, meaning that the directory `train/0/` contains all images that belong to class `0`.

Additionally, you'll find a `gt-train.tsv` and `gt-test.tsv` in the top level directory, which is a Tab Separated Values (TSV) file with two columns. The first column is the relative path from the location of the TSV file to the image and the second column is the class it belongs to. This is purely for convenience and you decide whether you want to load the data based on the file structure or use the TSV files as an index.

Note: Do ***not*** touch the test set during training or to decide which hyperparameters are the best. It is purely used at the end to evaluate the best models you have chosen. In order to determine the best model, you need to split off a validation set from the *training set*.

a) Support Vector Machine (SVM)

Build an SVM to train on the provided training set and classify the test set with it. Investigate at least *two* different kernels (e.g. linear and RBF) and optimise the SVM hyperparameters, such as C, by means of cross-validation.

b) Multilayer Perceptron (MLP)

Build an MLP with one hidden layer to train on the provided training set and classify the test set with it. Optimise the hyperparameters to obtain the best possible accuracy. The following hyperparameters should be considered:

- Hidden size (number of neurons), in the range $[10, 256]$
- Learning rate, typically in the range $[0.0001, 0.1]$
- *Optional:* Number of layers (one layer was asked, but you can try multiple layers as well)

c) Convolutional Neural Network (CNN)

Build a CNN to train on the provided training set and classify the test set with it. Optimise the hyperparameters to obtain the best possible accuracy. The following hyperparameters should be considered:

- Kernel size, most common: $\{3, 5, 7\}$
- Number of convolutions (layers)
- Learning rate, typically in the range $[0.0001, 0.1]$

The output of the CNN will be a feature map of size $C \times H \times W$. You need to reduce it to a vector of size equal to the number of classes, i.e. each element can be interpreted as a probability that it belongs to this class. A simple solution is to flatten the 3D output into a 1D vector and then apply a linear classifier (same as you did for the MLP).

Optional: Experiment with different feature aggregation strategies to feed them into the classifier, such as flatten or various forms of pooling.

Expected Submission

- Access to your Git(Hub) repository so that we can inspect your code.
- Short report in your Git repository (Markdown or PDF) with your results:
  - For SVM: Loss and accuracy during cross-validation for at least *two* kernels (e.g. linear and RBF) and tested hyperparameters, such as C.
  - For MLP and CNN: Plot of the loss and accuracy on the training set and validation set with respect to the training epochs with the different hyperparameters.
  - Accuracy on the test set of the best models for SVM, MLP and CNN, respectively.

For the individual task, calculate the forward pass of the CNN and hand in your solution on ILIAS `Exercises/Individual Exercises/2-CNN`.