# Practical 3: Artificial Neural Network

Contributing Students:
Barış Özakar

January 8, 2021

## Preparation of data

For this practical we are using the MNIST dataset, which contains handwritten digits 0-9. Therefore, our labels are categorical. After doing the appropriate reshapings, we normalized our data by dividing the pixel values with 255. In order to appropriately decide on the hyperparameters and do training with validation, we split our dataset into training, validation and test sets containing up 42000, 18000, 10000 images respectively.

## Building the model

Our Convolutional Neural Network consists of two convolution layers of 32 and 62 (3x3) filters with max-pooling and dropout layers in between. The fully connected layers consists of two Dense layers of 128, 64 and 10 neurons with batch normalization and dropout(0.25) layers in between. We have added batch normalization and dropout layers in order to overcome overfitting because without them, we were faced with a case of overfitting where our model was not able to generalize well. We have tried many combinations of layer numbers, but empirically, this combination yielded the best accuracy in the validation and test sets.

## Training

Since our data is categorical, we have used the categorical cross entropy loss function. In terms of optimizers, we have experimented with SGD and Adam optimizers, however we finally empirically decided on Adam as it converged faster with a learning rate of 0.001. We have tried other learning rate values (0.01 and 0.0001) however, learning rate of 0.001 had the best convergence properties. We used a batch size of 32 with 10 epochs and our training and validation set accuracies were %98.58 and %98.98 respectively. Upon testing our model on the test set, we have achieved an accuracy of 99.20%, which is above the benchmark expectations of 97%.

# DIFFICULTIES

Long training times and getting the accuracy over 97% was among the main difficulties that we faced. In order to solve long training times, we have decided to use Google Colab by running on the GPU so that training was faster. Getting the accuracy over 97% took empirical observations and changes to our model and our hyperparameters (addition of dropout and batchnormalization layers, determining a learning rate that is suitable, adding the right amount of convolution layers/filters and dense layers/neurons to allow for successful decision making).