# Maria Musiał 156062 – Analysis of a model II

Model Is designed to classify handwritten digits.

It uses fully connected CNN with 3 hidden layers.

* Input: image channels
* 2 convolutional layers +relu + maxPooling
* 3 fully connected hidden layers
* Output layer (size 10- number of classes)

We have 60000 images in train set and 10k in test. It is enough for a data of this complexity to have good model accuracy.

Data has 1 channel and its binary values.

Weights are initialized withing standard deviation of 0.01

**Breakdown of layers:**

conv1 -> relu1 -> max\_pool\_1 -> conv2 -> relu2 -> max\_pool\_2 -> flatten -> fc1 -> relu -> fc2 -> relu -> fc3 -> output

First layer is convolution with 20 filters. This should extract easy patterns like edges. Then we apply ReLU after adding bias vector. Next, we perform pooling (reducing idmensions, while retaining most important features in the image.

Next, we have convolution layer with 30 filters. It extracts more important features from image. Apply ReLU and max-pooing with size 2.

Then, we flatten 3D tensor we got to 2D tensor. Each row represents one image, and one column represents a flattened feature.

Now, we have 3 fully connected layers with ReLU as activation. [we have matrix multiplication of images and weights, addition of bias and ReLU] First layer has 1024, second 256 and third 64 neurons.

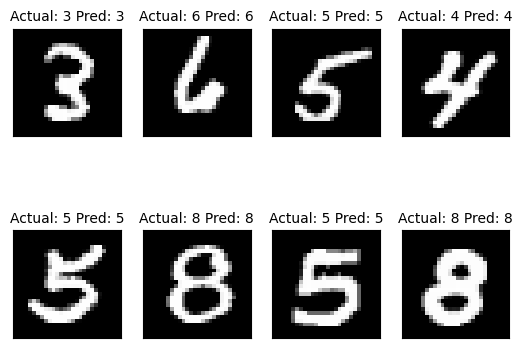
For output layer we use softmax (as we have classification. Probabilities of images should add up to 1)

For loss function we are using Adam optimizer with LR 0.001, beta 0.9, beta2 0.999 minimizing the sparse softmax cross entropy with logits

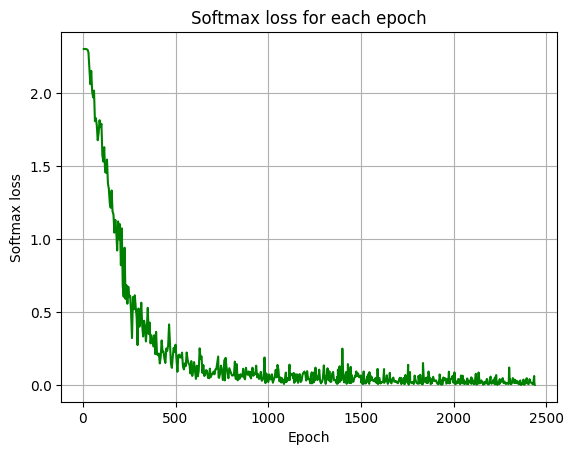
# **RESULTS**

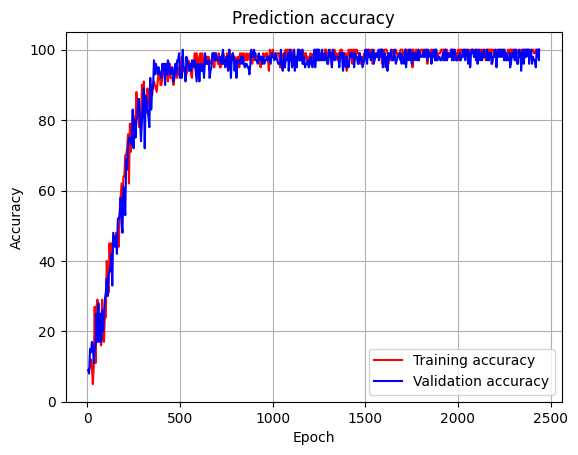
It took 6 minutes,- 2438 epochs to train it. Stopping condition was loss function under 0.001.

Some classified examples: (from test data)



Validation curves:





We can say that loss function looks like its supposed to. No overfitting, no underfitting. The fluctuations in line are due to complicated nature of data (images) .

The same goes for prediction accuracy. The model quite quickly converges to satisfactory values.

**MLP vs CNN**

CNN:

Architecture:

* Convolutional layers ( extracts complex features, spatial hierarchies, local patterns by convolution )
* Pooling layer (reduce spatial dimensions while preserving important information)
* Fully connected layers (process the flattened results of convolution to classify)

Spatial awareness:

* Local features can be detected thanks to convolutional layers
* With each layer CNN can detect more complex patterns

MLP:

Architecture:

* input is a vector, there are 1 or more dense layers with activation functions (1 neuron)

Spatial awareness:

* MLP lacks spatial awareness, as it works on entire vector at once. It wont capture complex features.