**Neural Network 1.0 Readme:**

**1) Import training data and generate new network**

This option generates a new neural network based on training images. Number of input neurons is equal to the amount of pixels of the images and number of output neurons is determined by the number of symbols for recognition.

Training images are located in the folder “training\_data”. The name of the image file must contain the symbol it shows (expected output). There are currently 4 symbols: square, circle, triangle, and cross. For each symbol 24 training images are provided. The imaged must be in the following format (no checks provided yet):

- squared (here: 25x25 pixels = 625 input neurons)

- windows bitmap format, uncompressed (.bmp)

- 24-bit RGB (internally converted to grayscale)

After images are imported, number of hidden layers and neurons per hidden layer must be entered. **It is not recommended to choose more than 4 hidden layers as training time quickly increases!**

**2) Import model and generate saved network**

A network can be recreated from a model file located in the folder “models”. A model is a pretrained network where number of layers/neurons per layer and weights/biases are saved in a file (.mdl). Importing a model also creates the appropriate network.

**3) Train network with training images**

To train an existing network on training images, **menu option 1) must first be performed**! During training, the average cost of the training set for each iteration of the backpropagation algorithm is shown. Its purpose is to minimize the cost function using gradient descent. The function stops if either the cost is below 0.001 **OR** if 10.000 iterations are done (emergency exit of the loop in case cost can’t be minimized below desired threshold). A learning rate of 1.0 is preconfigured. All three parameters are defined in “network.hpp” and can be adjusted.

**4) Test network with test images**

After training is finished, the network can be tested with unknown pictures **which were not part of the training set**. Test images must be placed in the folder “test\_data”. Single images can be selected for testing. Output of the four labeled output neurons is displayed and the probability that a symbol is recognized. If a square image is chosen, the output should look like this:

circle: 0.990

square: 0.004

cross: 0.000

triangle: 0.025

Cost: 0.000748881

In addition, the cost of the image is shown, which is only for test purposes and not even possible for unlabeled images.

**5) Export model**

A trained network can be exported. This process saves the network parameters including all weights and biases to a file with the extension .mdl in the folder “models”. The name of the file can be chosen.

**6) Reset network**

Resets all parameters of a network and free allocated memory to import or create a new network.

**7) Print network**

Prints all network parameters. This is for testing and debugging purposes and is not recommended for larger networks.

**8) Print training data**

Prints the vectors generated by import of test images (inner representation of images). This is for testing and debugging purposes and is not recommended for larger images/ image sets.

**9) Exit program**