MATLAB Neural Network Toolbox Tutorial

1. Prepare data

(1) The dataset named “MNIST_subset” is available on Moodle for downloading. The training set and testing set are ready for you to use.

(2) The label should be represented using a one-hot representation scheme, for example, the digit “1” could be denoted as [0100000000] if you have 10 digits all together. Each class label should have a distinct form of representation.

(3) Please use MATLAB’s built-in Neural Network Toolbox (NNToolbox) to complete this task. Just type “nntool” in your command window and see what it is like.

2. Create network using nntool

Make sure the data is in the correct format, as specified above. Next, run the nntool command. This will open a Graphical User Interface (GUI), see Fig. 1.

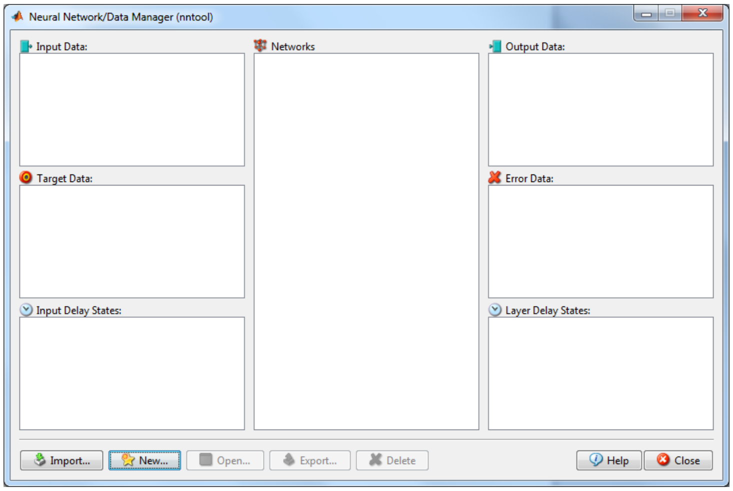
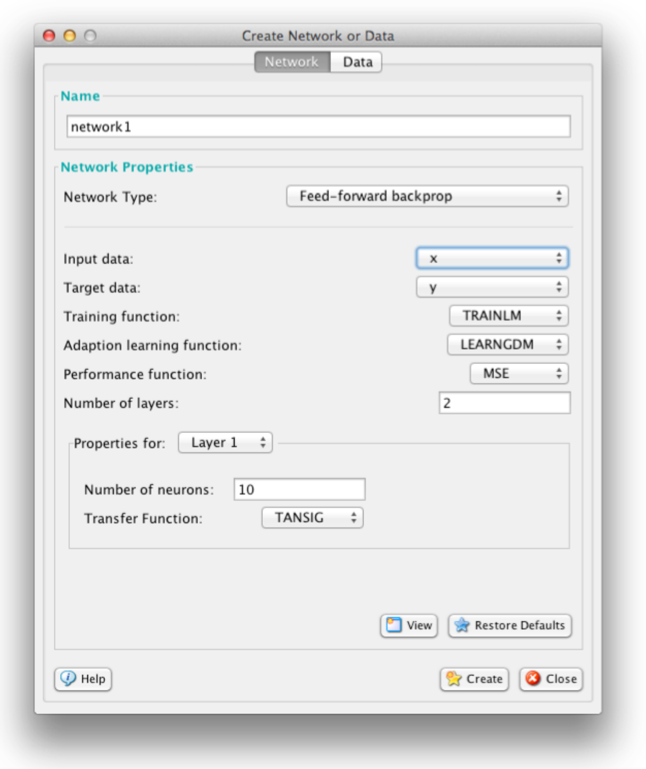


Figure 1: The main window of the *nntool* Neural Networks GUI.

Import the training data as inputs and as targets using the 'import' button. Make sure you import the MNIST training sample as 'inputs' and the training labels as 'targets'. Next create a new network by clicking the ‘New...’ button. This will bring up the ‘Create Network or Data’ window, see Fig. 2.



Create Network or Data

Name: network1

Network Properties

Network Type: Feed-forward backprop

Input data: x

Target data: y

Training function: TRAINLM

Adaption learning function: LEARNGDM

Performance function: MSE

Number of layers: 2

Properties for: Layer 1

Number of neurons: 10

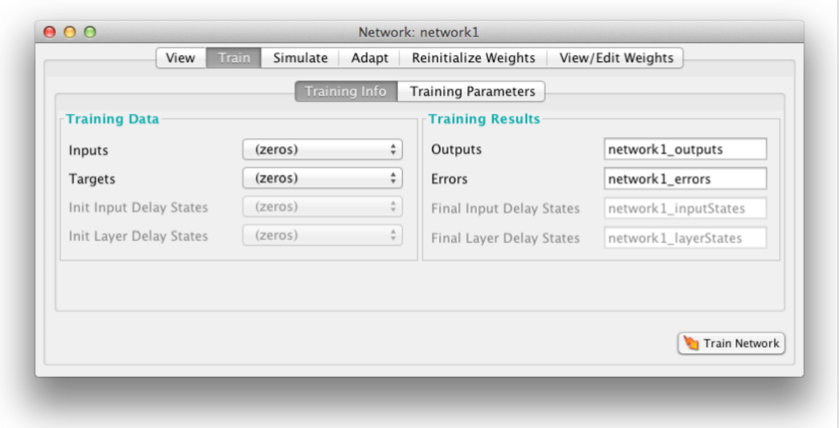
Transfer Function: TANSIG

Figure 2: The 'Create Network or Data' window.

Select Feed forward backprop from the network type list. Choose your features as the input data and your labels as the target data. The training function should be one of the backpropagation algorithms. If you want to change the number of layers, make sure to hit 'enter' after changing the value in the text box. Please note that the last layer you define is the output layer and as such should have the same number of neurons as the number of categories in which you want to classify your data. Click 'Create' when you are done.

Next select the network you've just created in the main window of the nntool and press the 'Open ...' button. This will bring the network visualisation window. Go to the 'Train' tab, where you can set the training parameters, see Fig. 3.

Specify the training input and targets. Be aware that some training functions are very slow, so don't start training with too many epochs (100 should be fine). You can set the value of the epochs in the 'Training Parameters' tab. Here you can also set how often the graphical output of training info is given. Set this value low, say every 5 or 10 epochs. When you feel confident that the NN is learning fast enough, you can increase the number of epochs if you wish. What are the influences of the learning rate and of the number of epochs used?



Network: network1

Training Data

Inputs	(zeros)
Targets	(zeros)
Init Input Delay States	(zeros)
Init Layer Delay States	(zeros)

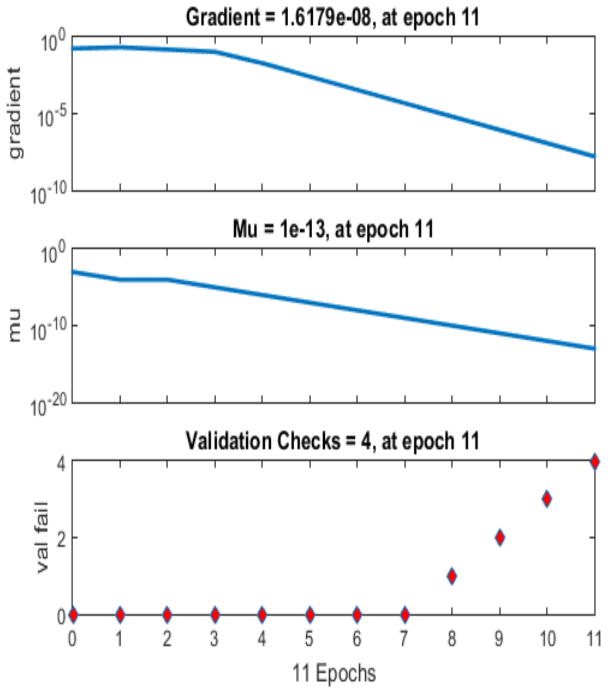
Training Results

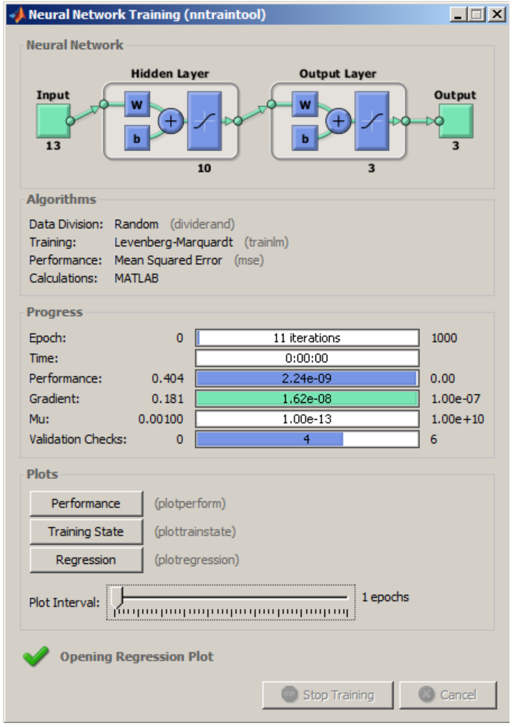
Outputs	network1_outputs
Errors	network1_errors
Final Input Delay States	network1_inputStates
Final Layer Delay States	network1_layerStates

Figure 3: Setting the network's training parameters.

3. Evaluation

Now that you have a basic understanding of Neural Networks by using the MNIST dataset from the previous assignment. You can evaluate the neural networks by inspecting some evaluation metrics after you've trained your neural network. Figure 4 shows an example result of evaluation.





Neural Network Training (nntraintool)

Algorithms

Data Division: Random (dividerand)
Training: Levenberg-Marquardt (trainlm)
Performance: Mean Squared Error (mse)
Calculations: MATLAB

Progress

Epoch:	0	11 iterations	1000
Time:		0:00:00	
Performance:	0.404	2.24e-09	0.00
Gradient:	0.181	1.62e-08	1.00e-07
Mu:	0.00100	1.00e-13	1.00e+10
Validation Checks:	0	4	6

Plots

Performance (plotperform)
Training State (plottrainstate)
Regression (plotregression)

Plot Interval: 1 epochs

✓ Opening Regression Plot

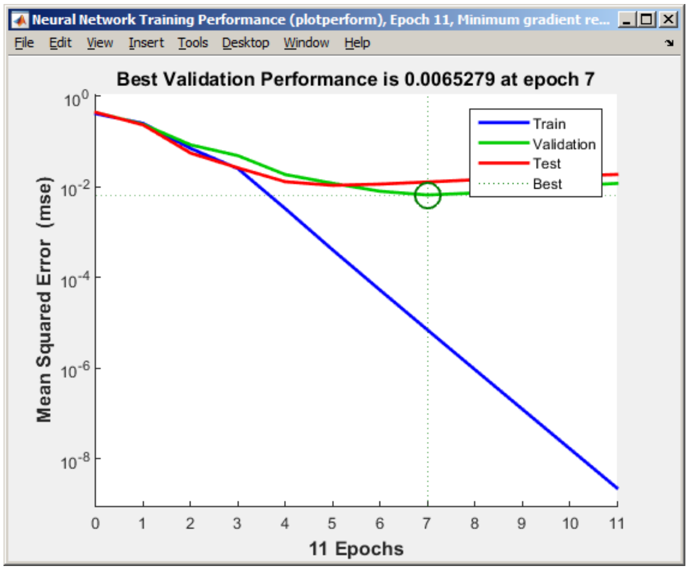


Figure 4. Performance evaluation example

NOTE that you could “Simulate” option (see Figure 3) to fit your trained network for unseen data. For example, you could use training set in MNIST_subset to construct your network and testing set to test the performance of your network.