Training Neural Network for Image Recognition with Neuroph Studio

Neuroph Studio provides environment for creating and training neural networks, which can be saved as ready-to-use java components. Also it provides specialised image recognition tool to train neural networks for image recognition. Creating and training neural network for image recognition consists of the following steps:

1. Create Neuroph project
2. Create image recognition neural network
3. Train network
4. Test network
5. Save & deploy network

Step 1. To create Neuroph Project click **File > New Project**

Enter project name and location, click **Finish**.

Step 2. Next, to create image recognition network, click **File > New File**.

Select Image Recognition file type, and click **Next**.

Next, choose images you want to be recognized, by selecting individual image files or by adding whole image directory. You can also do the basic image editing like cropping and resizing, by opening simple image editor with edit button.

You can use image recognition in full color mode or in binary black and white mode. The binary black and white mode represents pixel as [0, 1] and so it uses less number of input neurons. For some applications (like character recognition for example) binary black and white mode may be optimal solution.

Choose **Black and White** for now.

In next step choose image that should not be recognized, which will help to avoid false recognition. Usually these are blocks of all red, all green and all blue images, but also might include others.

When you test your image recognition network, you'll figure out what makes sense to include here.

Then, enter Training Set Label and Image Sampling Resolution, and click **Next**.

For start, you can use the default settings (20x20 resolution), and just provide the images.

The next thing to do, is to create the neural network.

Network label - The label for the neural network, which is usefull when you create several neural networks for the same problem, and you're comparing them.
Transfer function - This setting determines which transfer function will be used by the neurons. In most cases you can leave the default settings 'Sigmoid', but sometimes using 'Tanh' can give you better results.

Hidden Layers Neuron Counts - This is the most important setting which determines the number of hidden layers in network, and number of neurons in each hidden layer. Hidden layers are layers between input and output layer. The trick is to have the smallest possible number of layers and neurons which can succesfully learn the training set. The smaller number of neurons - the faster learning, better generalization. Suitable number of hidden neurons also depends of the number of input and output neurons, and the best value can be figured out by experimenting. For start, try 8x8 images and one hidden layer with 12 neurons, which is the default setting. If you wany to increase number of neurons, just enter the number for example '12' neurons. If you want to add more than one layer of neurons enter the number of neurons in each layer separated with space. For example, if you enter '12 8 6' it will create three hidden layers with 12, 8 and 6 neurons.

Click the '**Finish**' button to create the neural network. After you click the button new window with created neural network will open.

To train the network first double click on the network object. Then select the training set from project tree, click the rest button to reset the weight values and then click randomize to set new initial values. Then click the '**Train**' button. Leave the default parameters and proceed.

This will start training and open network learning graph and iteration counter, so you can obesrve the learning process. If the learning gets stuck (total network error does not go down), you can try with different number of neurons, layers or learning parameters. For learning rate and momentum use the values between [0, 1] , and for the error some small value bellow 0.1 is recommended. Some rule of the thumb values are 0.2 for learning rate and 0.7 for momentum.

After you have trained the network you can try how it works in the test panel. Select **Window > Image Recognition Test**. Select the network from the working directory. Click '**Select Test Image**' button to set input image for the network, and the network output will be displayed as the list of image labels and corresponding neuron outputs. The recognized image corresponds to the neuron with highest output.

To save the neural network as Java component click [Main menu > File > Save] and use the .nnet extension. The network will be saved as seralized *MultiLayerPerceptron* object.

** Original Source Material: http://neuroph.sourceforge.net/image_recognition.html