GETTING STARTED WITH NEUROPH

by Zoran Sevarac and Marko Koprivica

This guide gives you a brief overview on how to use *Neuroph* framework ***version 2.0.0 alpha***.

CONTENTS

1. What is Neuroph?

Neuroph is Java application framework for neural network development.

2. Whats in Neuroph?

Neuroph consists of set of Java classes and GUI application easyNeurons.
You can use Java classes to create NN in your Java programs, and application for experimenting with common neural network architectures.

3. Requirements

In order to use/run *neuroph* you need Java VM 1.6 installed on your computer.

4. Installation and Starting

Neuroph does not need any specific installation procedure. Just unpack downloaded neuroph_xxx.zip where you want and use it.

After unpacking you run demo application by left clicking on easyNeurons.jar
(if .jar extension is not associated with Java you can right the easyNeurons.jar
click then and select: Open With>Java (TM) Open Platform SE binary)

You can also start demo application from command line by typing:

java –jar easyNeurons.jar

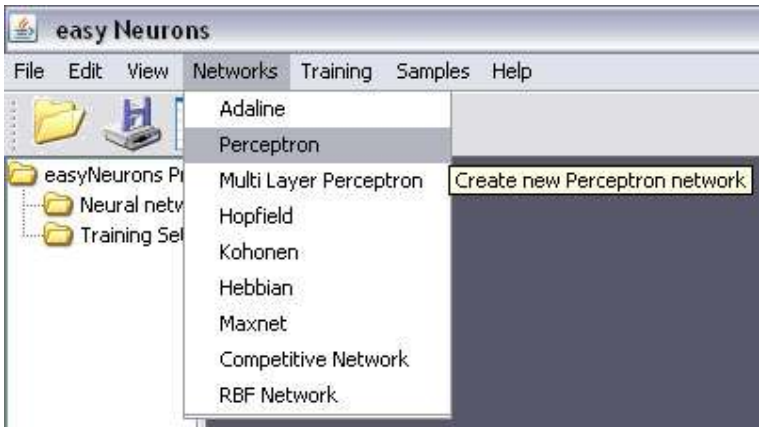
5. Training neural network with easyNeurons application

Now we'll explain how to use application easyNeurons to create neural networks. There are 5 steps for training NN, and they will be described with example Perceptron neural network for logical OR function (V).

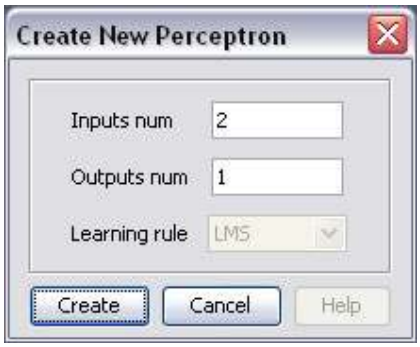
To create and train neural network with easyNeurons do the following:

1. Choose NN architecture (in main manu choose Networks>Perceptron)
2. Enter architecture specific parameters (eg. num of neurons)
3. Create training set
4. Set training parametars and start training
5. Test network

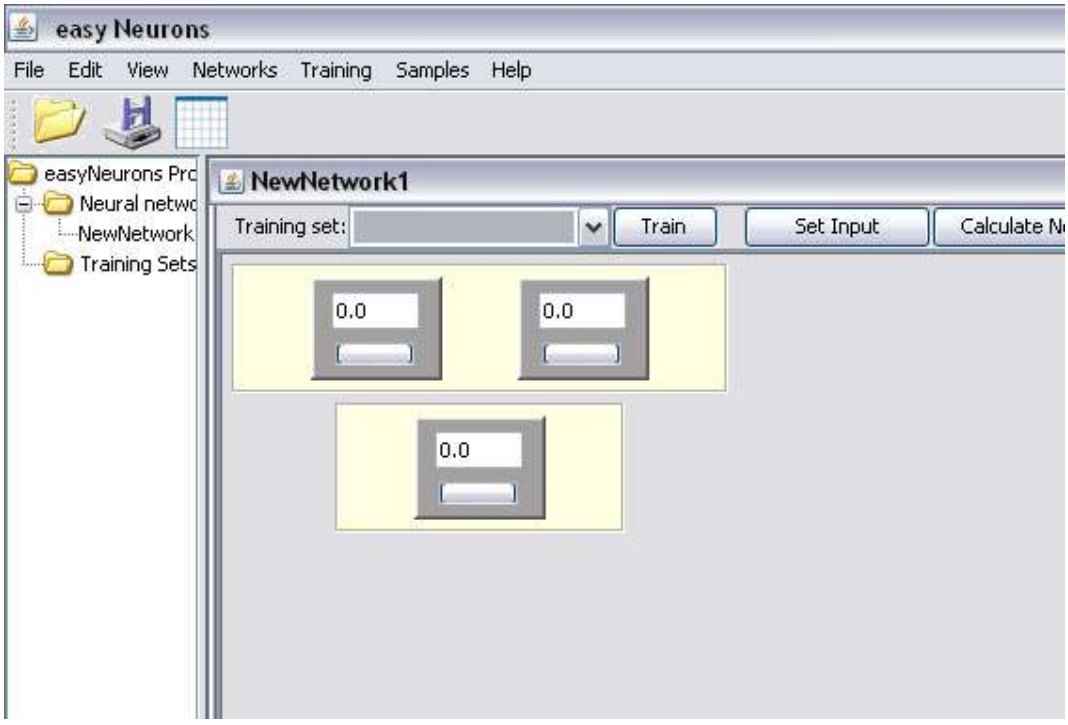
Step 1. To create Perceptron network, in main menu click ***Networks > Perceptron***



Step 2. Enter number of neurons in input and output layer, and click ***Create*** button.

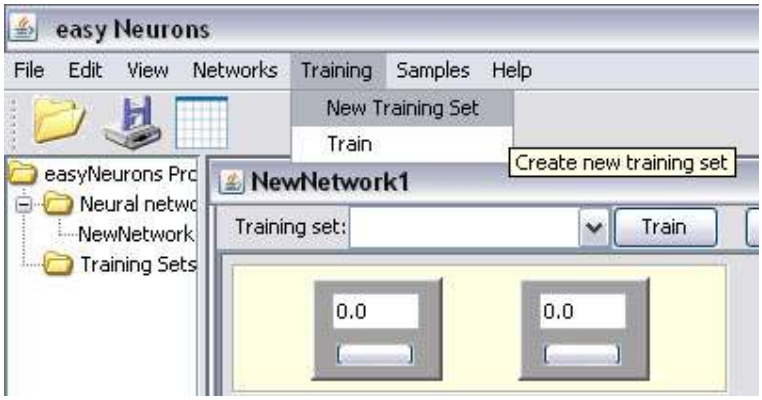


This will create the Perceptron neural network with two neurons in input, and one in output layer. By default, all neurons will have **Step** transfer functions.



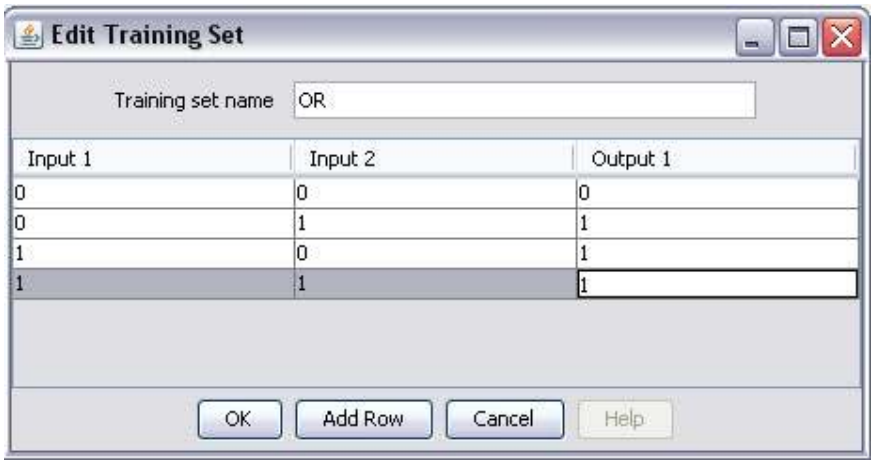
Now we shall train this simple network to learn logical OR function. First we have to create the ***training set*** according to OR truth table.

Step 3. In main menu click ***Training > New Training Set*** to open training set training set wizard

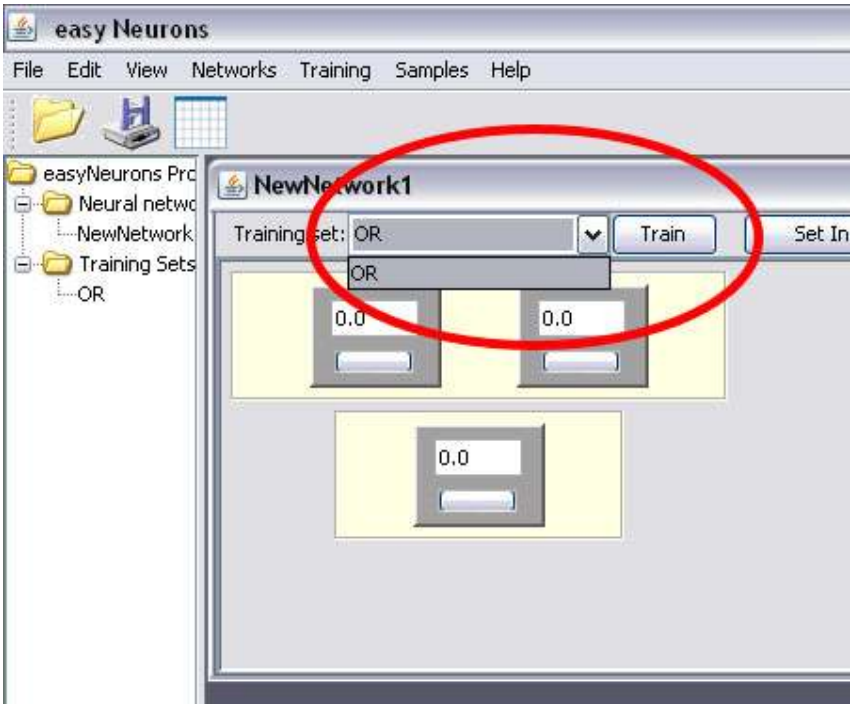


Enter training set name, number of inputs, number outputs as shown on picture below and click ***Create*** button.

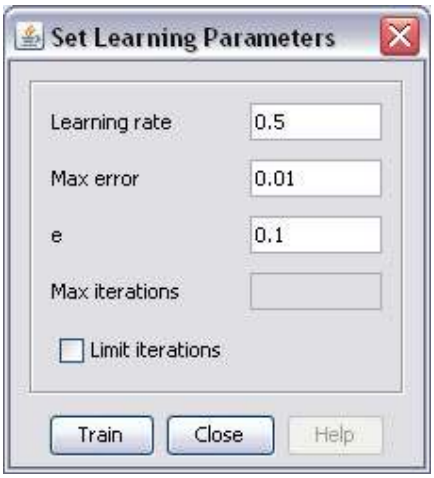
Then create training set by entering **training elements** as input and desired output values of neurons in input and output layer. Use ***Add row*** button to add new elements, and click ***OK*** button when finished.



Step 4. To start network training procedure, in network window select training set from drop down list and click ***Train*** button.

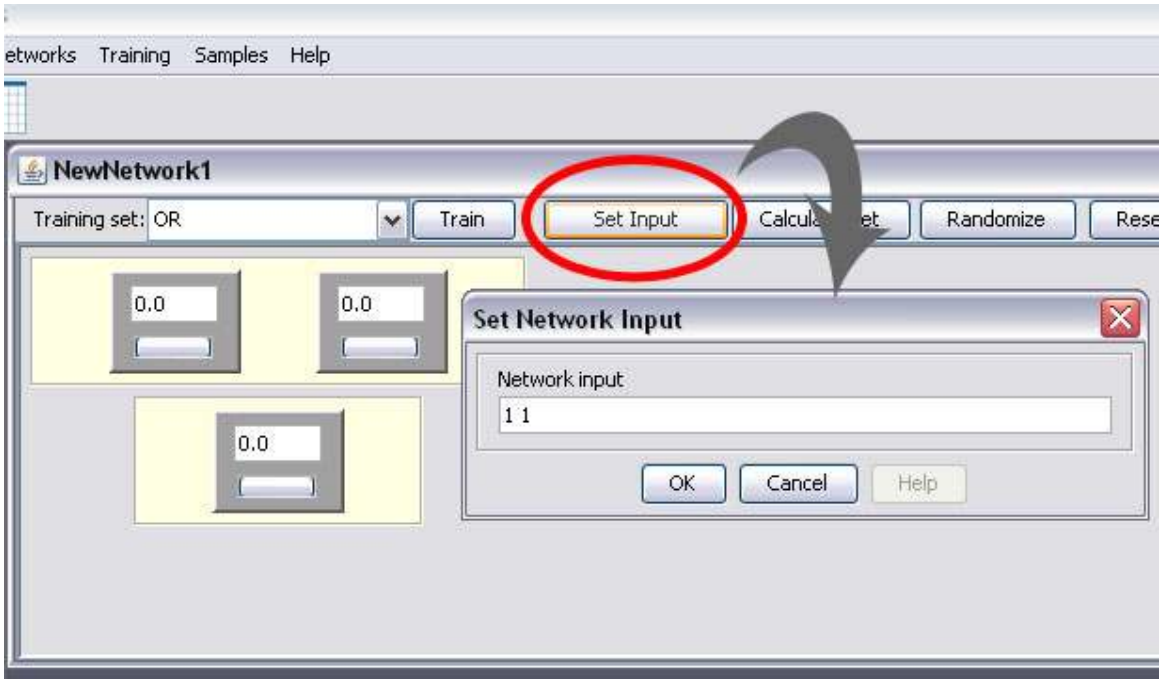


In ***Set Learning parameters*** dialog use default learning parameters, and just click the ***Train*** button.



When the Total Net Error is zero, the training is complete.

Step 5. After the training is complete, you can test network by using **Set Input** button. This opens ***Set Network Input*** dialog in which you can enter input values for network separated with whitespace.



6. Creating NN in Java code with Neuroph

This is the same example as in previous chapter, but now in Java code. Here is how to create, train and save Perceptron neural network with *Neuroph*:

```
// create new perceptron network
NeuralNetwork neuralNetwork = new Perceptron(2, 1, TransferFunction.STEP);

// create training set
TrainingSet trainingSet=new TrainingSet();

// add training data to training set
trainingSet.addElement(new SupervisedTrainingElement("0 0", "0"));
trainingSet.addElement(new SupervisedTrainingElement("0 1", "1"));
trainingSet.addElement(new SupervisedTrainingElement("1 0", "1"));
trainingSet.addElement(new SupervisedTrainingElement("1 1", "1"));

// learn the training set
neuralNetwork.learn(trainingSet);

// save the trained network into file
neuralNetwork.save("or_perceptron.nnet");
```

The following example shows how to use saved network.

```
// load the saved network
NeuralNetwork neuralNetwork = NeuralNetwork.load("or_perceptron.nnet");

// create some test input for network
Vector <Double> networkInput = new  Vector<Double> ();
networkInput.add(new Double(1));
networkInput.add(new Double(1));

// set network input
neuralNetwork.setInput(networkInput);

// calculate network
neuralNetwork.calculate();

// get network output
Vector <Double> networkOutput = neuralNetwork.getOutput();
```

This examples show the basic usage of neural network created with *Neuroph*. For more details about all available NN architectures, see the *Neuroph* API documentation.

7. Web Links

http://neuroph.sourceforge.net
http://en.wikipedia.org/wiki/Artificial_neuron
http://en.wikipedia.org/wiki/Artificial_neural_network