GETTING STARTED WITH NEUROPH

by Zoran Sevarac and Marko Koprivica

This guide gives you a brief overview on how to use *Neuroph* framework ***version 2.7***

CONTENTS

1. What is Neuroph?

Neuroph is Java framework for neural network development.

2. Whats in Neuroph?

Neuroph consists of the Java library and GUI neural network editor called Neuroph Studio.
You can experiment with common neural network architectures in Neuroph Studio, and then use Neuroph Java library to use those neural networks in your Java programs.

3. Requirements

In order to use/run *Neuroph* you just need Java VM 1.7 installed on your computer. Everything else is provided in downloaded package.

4. Installation and Starting

To install NeurophStudio just start the installer wizard (neurophstudio-windows.exe for windows or neurophstudio-linux.sh for linux) and follow the simple wizard steps.

Neuroph framework doesnt needs any specific installation procedure, just add reference to *neuroph-core-xx.jar* library to your Java project (and any other required jars like neuroph-imgrec-2.7.jar if you're using image recognition etc.) .

5. Training neural network with NeurophStudio application

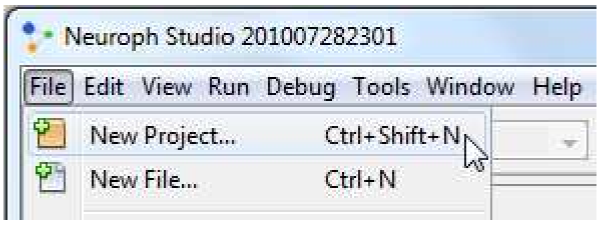
Now we'll explain how to use NeurophStudio to create neural networks. There are 5 steps for training NN, and they will be described with example Perceptron neural network for logical OR function (V).

To create and train Perceptron neural network using Neuroph Studio do the following:

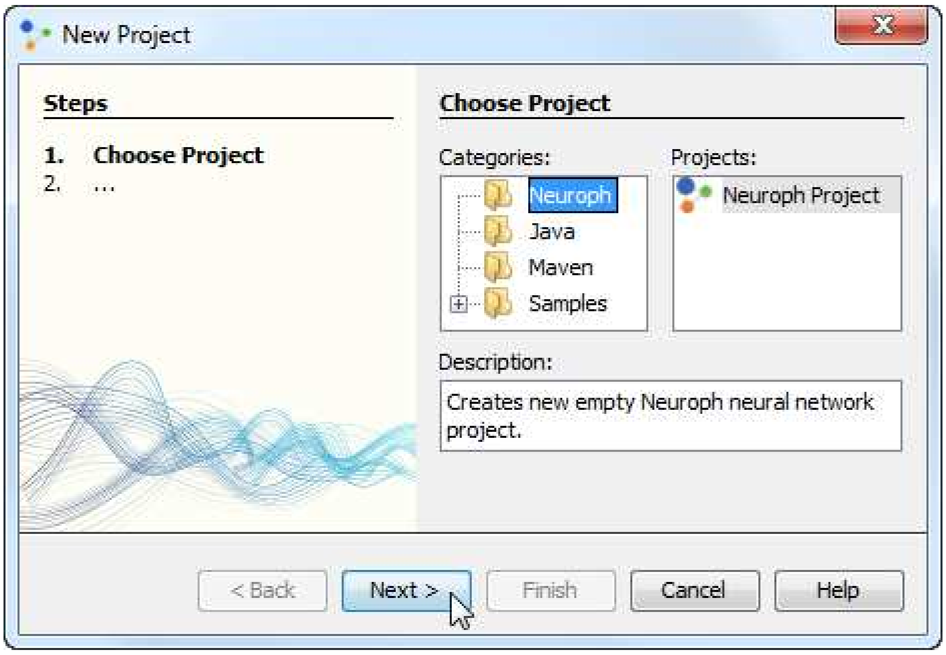
1. Create Neuroph Project.
2. Create Perceptron network.
3. Create training set (in main menu choose Training >New Training Set).
4. Train network
5. Test trained network

Step 1. Create Neuroph project.

Click File > New Project.


Neuroph Studio 201007282301
File Edit View Run Debug Tools Window Help
New Project... Ctrl+Shift+N
New File... Ctrl+N

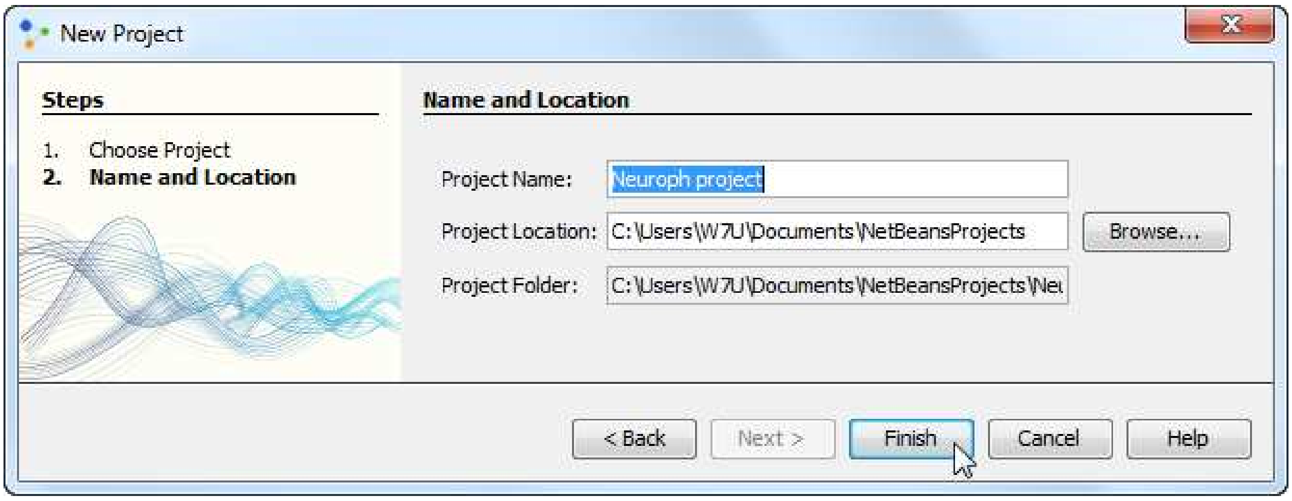
Then, select Neuroph Project and click Next.



Description:

Creates new empty Neuroph neural network project.

Enter project name and location, click Finish.



Project Name: Neuroph project

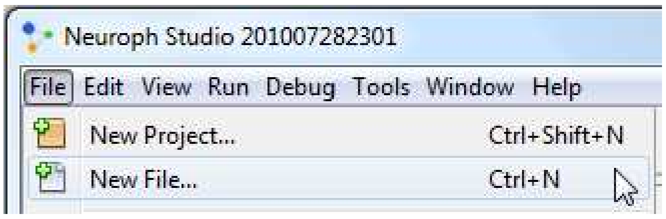
Project Location: C:\Users\W7U\Documents\NetBeansProjects

Project Folder: C:\Users\W7U\Documents\NetBeansProjects\Neu

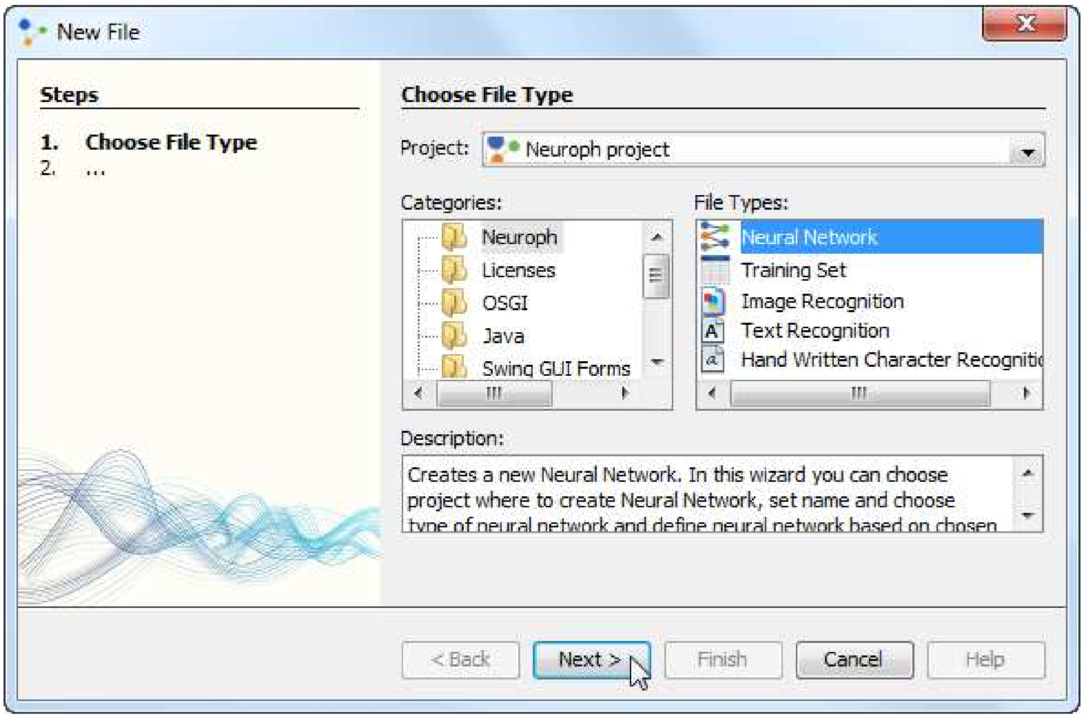
Project is created, now create neural network.

Step 2. Create Perceptron network.

Click File > New File

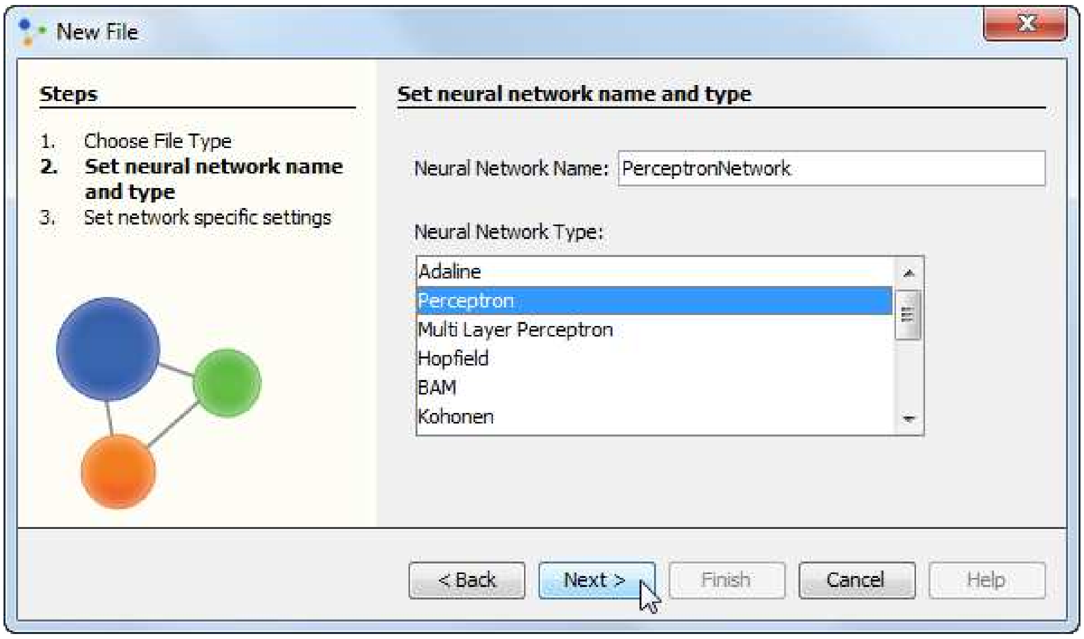

Neuroph Studio 201007282301
File Edit View Run Debug Tools Window Help
New Project... Ctrl+Shift+N
New File... Ctrl+N

Select desired project from Project drop-down menu, Neuroph as category, Neural Network file type and click next.



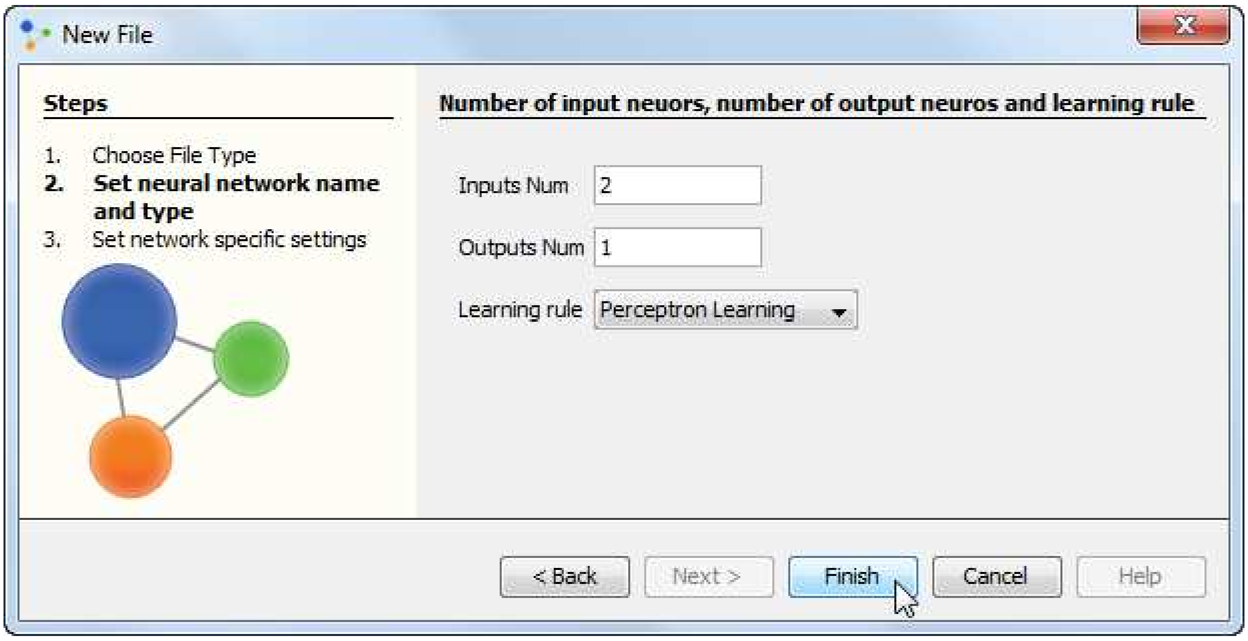
Creates a new Neural Network. In this wizard you can choose project where to create Neural Network, set name and choose type of neural network and define neural network based on chosen

Enter network name, select Perceptron network type, click next.



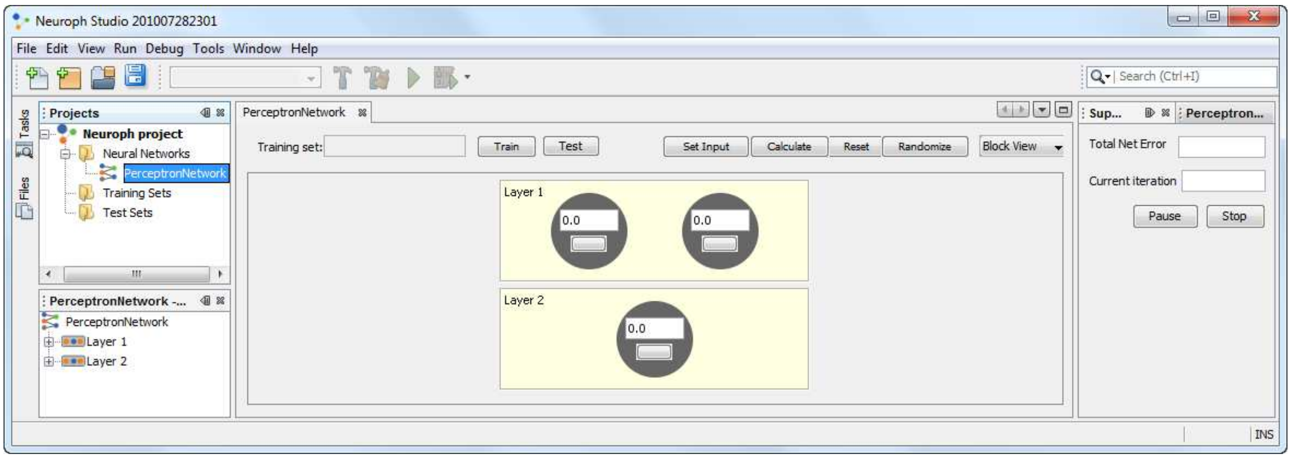
Neural Network Name: PerceptronNetwork

In new perceptron dialog enter number of neurons in input (2) and output layer (1) , choose Perceptron Learning and click Create button.

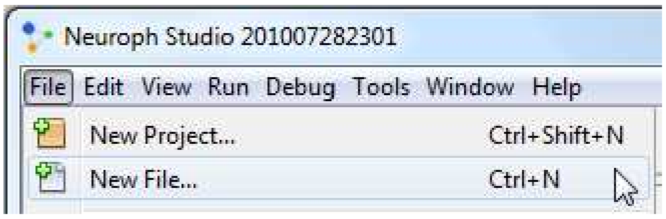

New File
Steps
1. Choose File Type
2. Set neural network name and type
3. Set network specific settings
Number of input neuors, number of output neuros and learning rule
< Back
Next >
Finish
Cancel
Help

Inputs Num: 2
Outputs Num: 1
Learning rule: Perceptron Learning

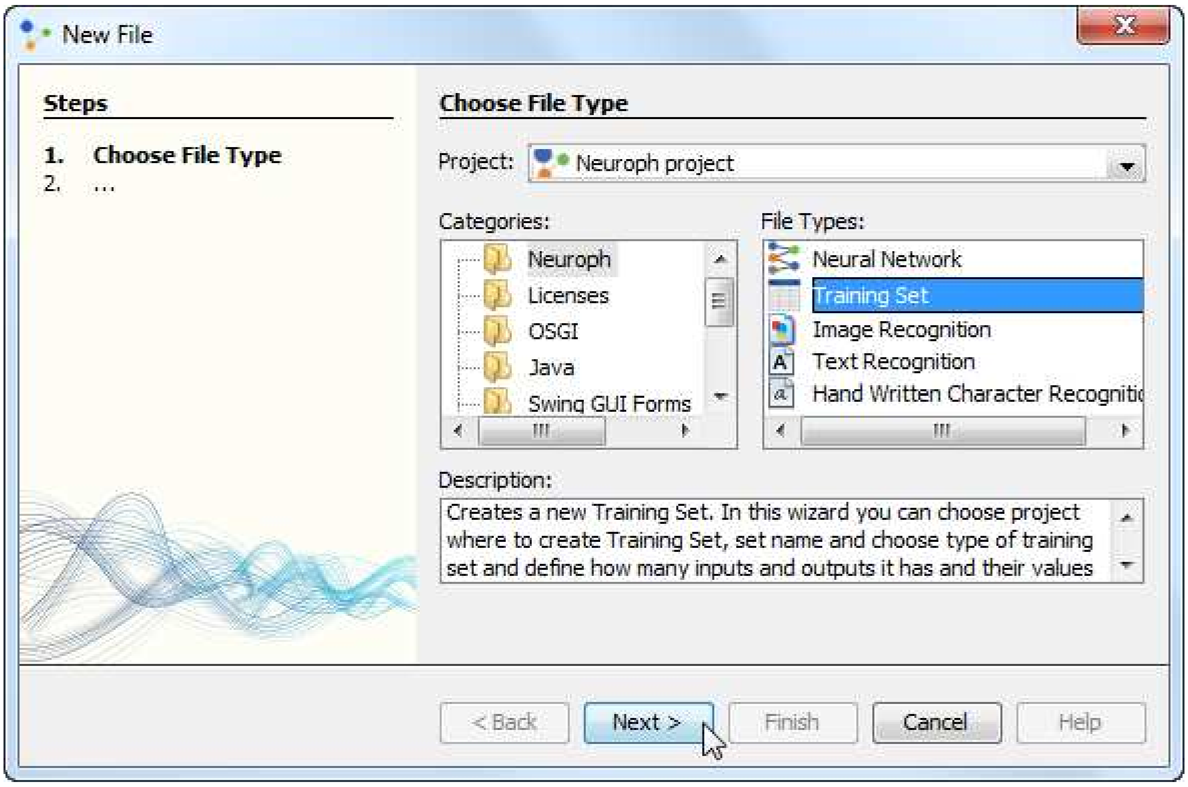
This will create the Perceptron neural network with two neurons in input, and one in output layer, all with Step transfer functions.



Step 3. To create training set, click File >New File to open training set wizard.


Neuroph Studio 201007282301
File Edit View Run Debug Tools Window Help
New Project... Ctrl+Shift+N
New File... Ctrl+N

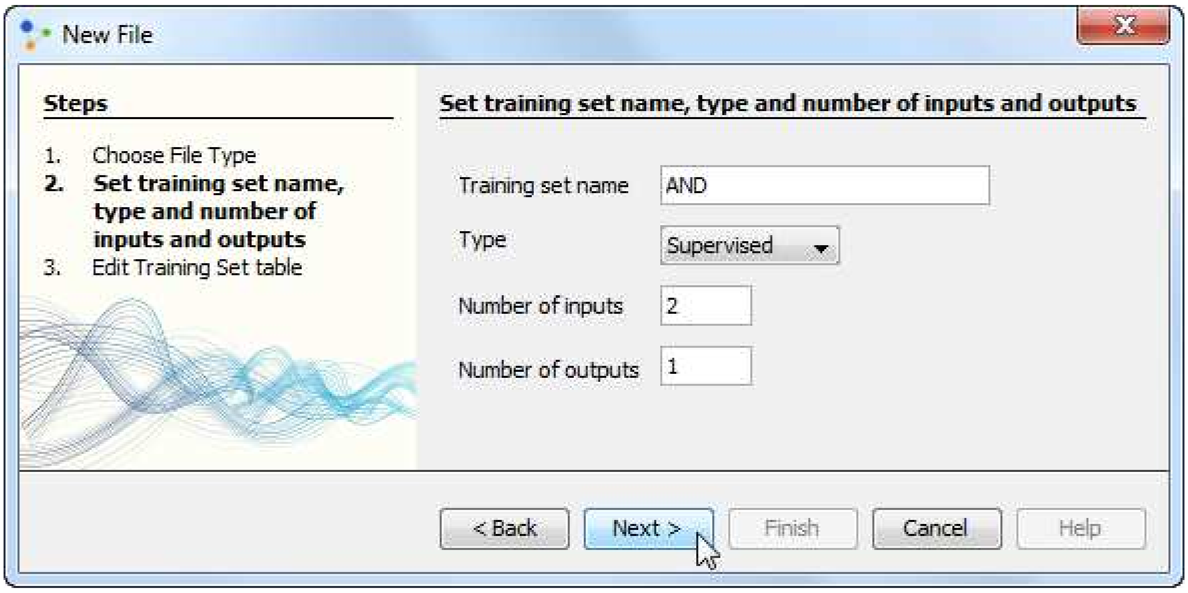
Select Training set file type, then click next.



Project: Neuroph project

Description:
Creates a new Training Set. In this wizard you can choose project where to create Training Set, set name and choose type of training set and define how many inputs and outputs it has and their values

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


New File
Steps
1. Choose File Type
2. Set training set name, type and number of inputs and outputs
3. Edit Training Set table
Set training set name, type and number of inputs and outputs
< Back
Next >
Finish
Cancel
Help

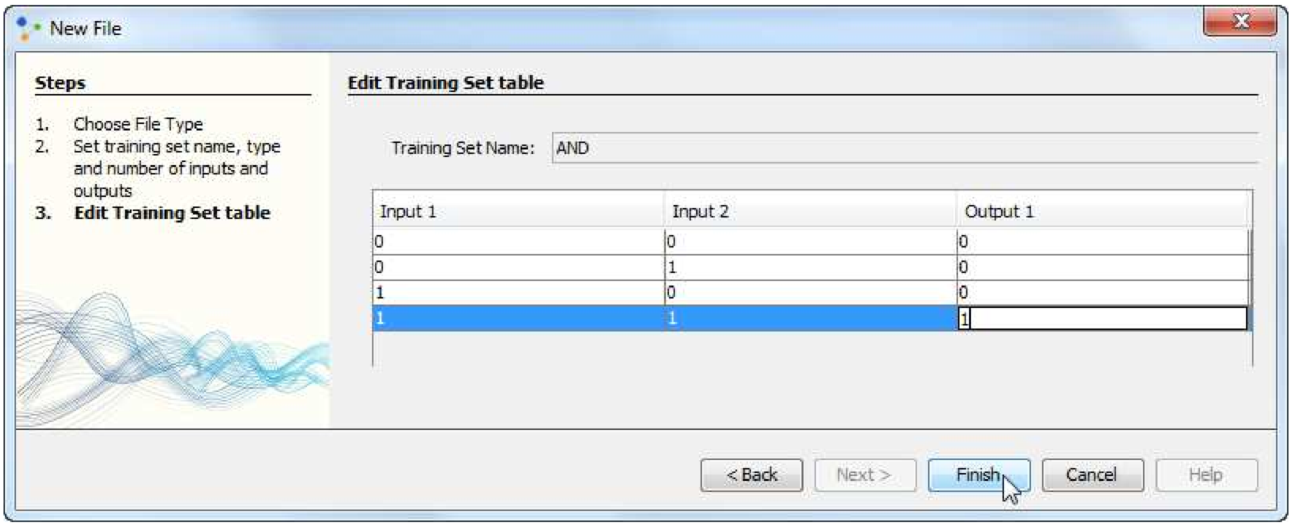
Training set name: AND

Type: Supervised

Number of inputs: 2

Number of outputs: 1

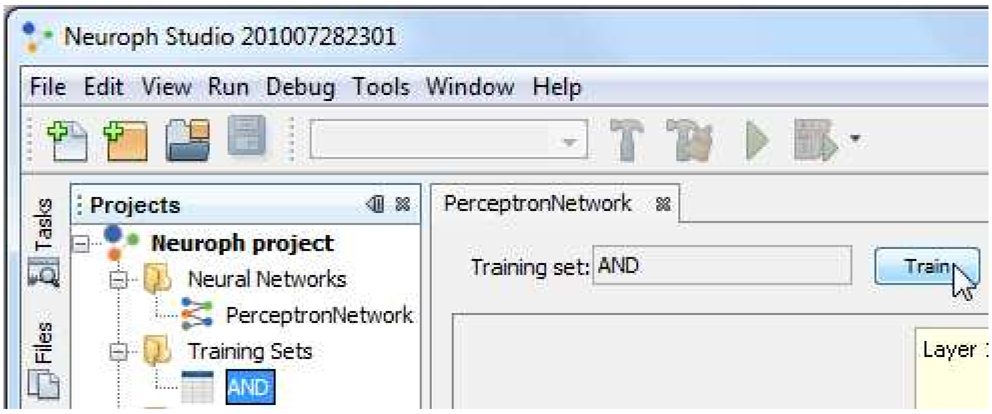
Then create training set by entering training elements as input and desired output values of neurons in input and output layer. Use Add row button to add new elements, and click OK button when finished.


New File
Steps
1. Choose File Type
2. Set training set name, type and number of inputs and outputs
3. Edit Training Set table
Edit Training Set table
< Back
Next >
Finish
Cancel
Help

Training Set Name: AND

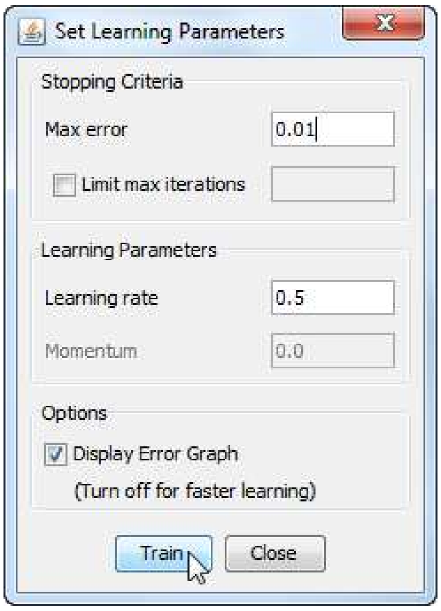
Input 1	Input 2	Output 1
0	0	0
0	1	0
1	0	0
1	1	1

Step 4. Training the network. To start network training procedure, in network window select training set and click Train button.



Training set: AND

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



Max error: 0.01

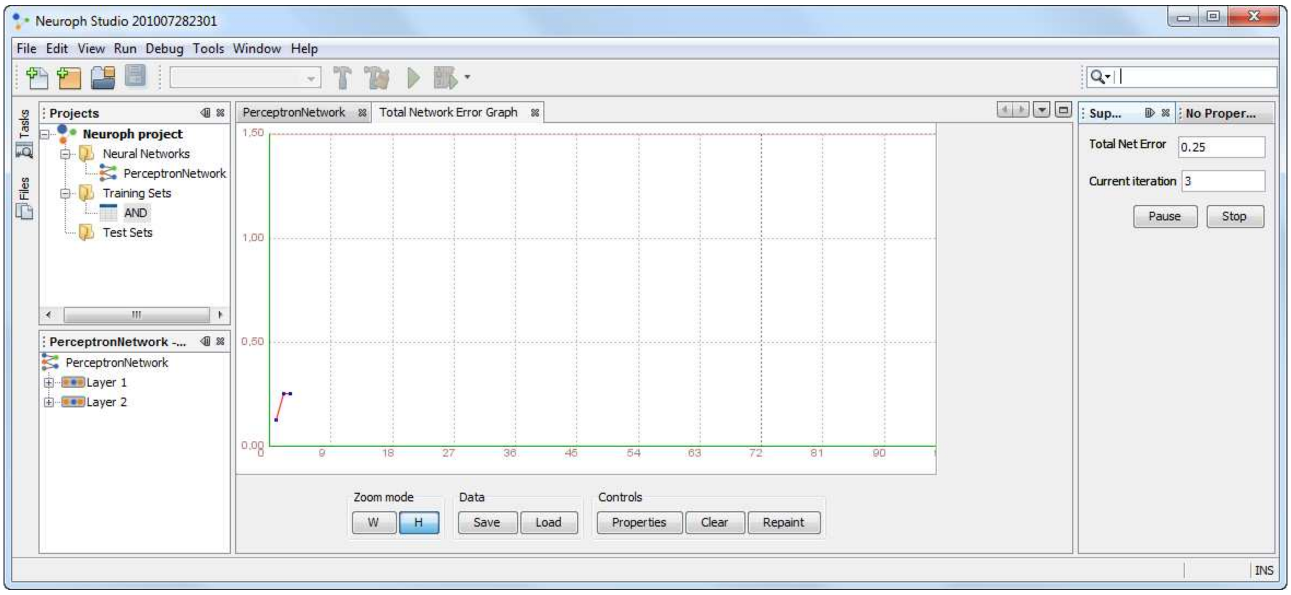
Limit max iterations: ☐

Learning rate: 0.5

Momentum: 0.0

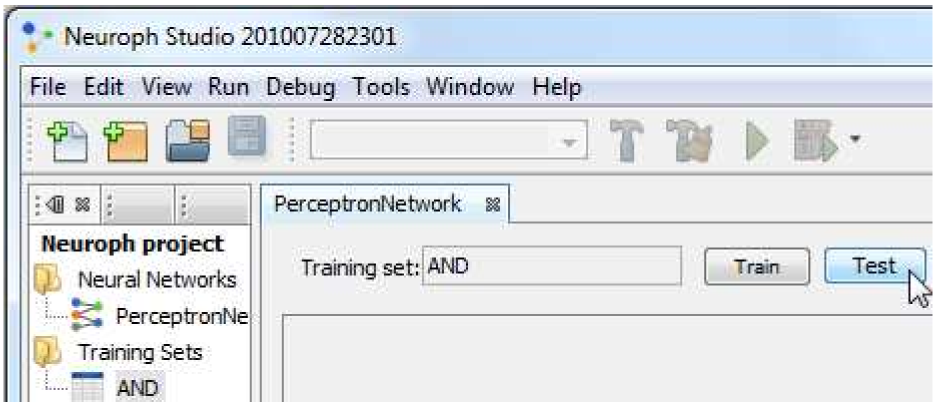
☑ Display Error Graph (Turn off for faster learning)

When the Total Net Error is zero, the training is complete (which will be very fast since this example is very simple)



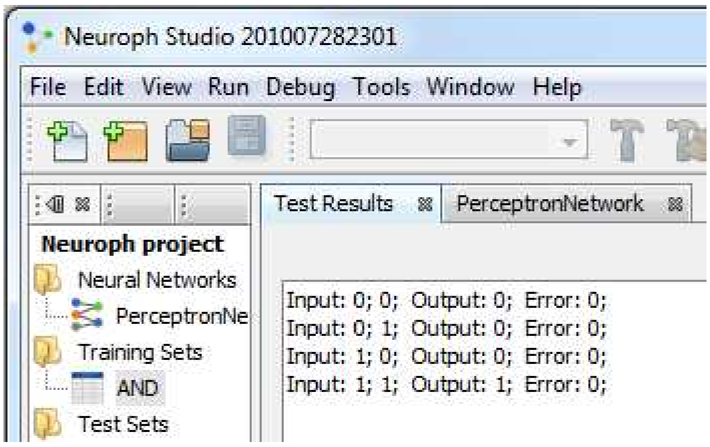
Total Net Error 0.25
Current iteration 3

Step 5. After the training is complete, you can test the network for the whole training set by selecting training set to test, and clicking Test button..



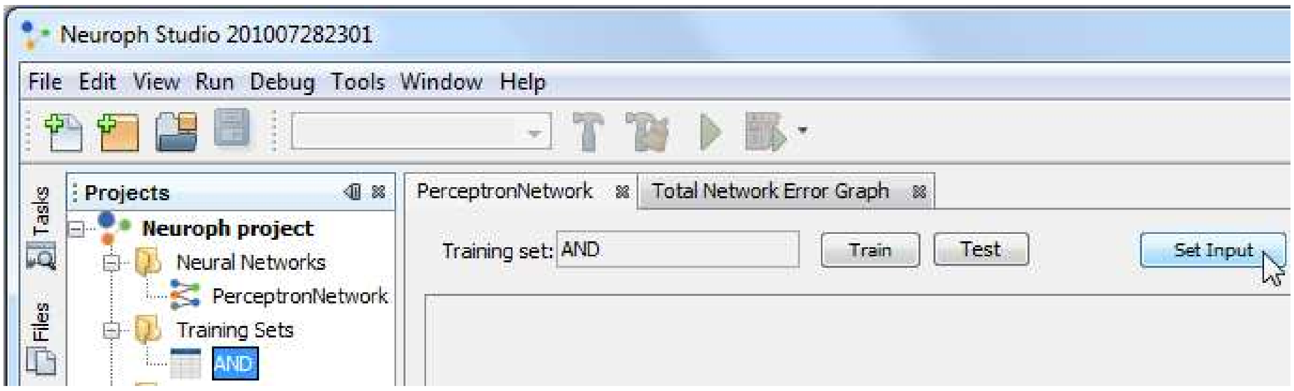
Training set: AND

This will open test results in a new tab.

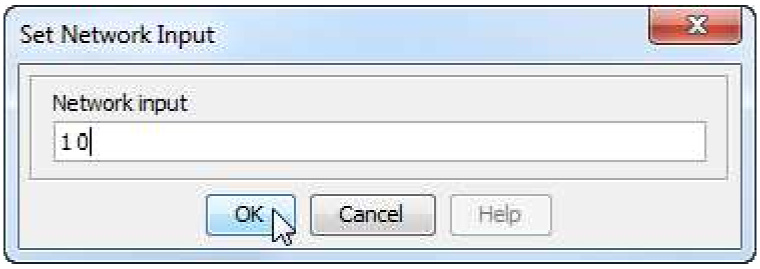


```
Input: 0; 0;  Output: 0;  Error: 0;
Input: 0; 1;  Output: 0;  Error: 0;
Input: 1; 0;  Output: 0;  Error: 0;
Input: 1; 1;  Output: 1;  Error: 0;
```

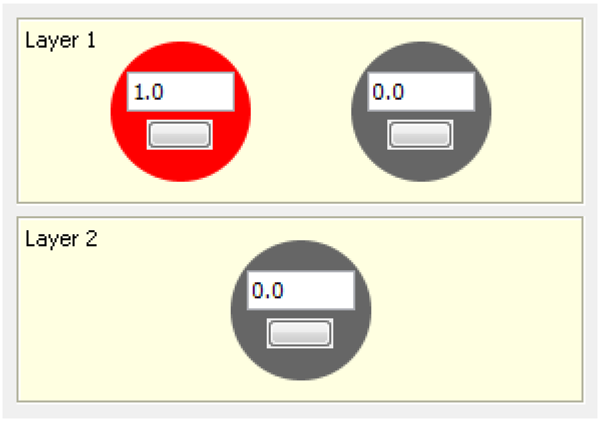
To test single input, use Set Input button.



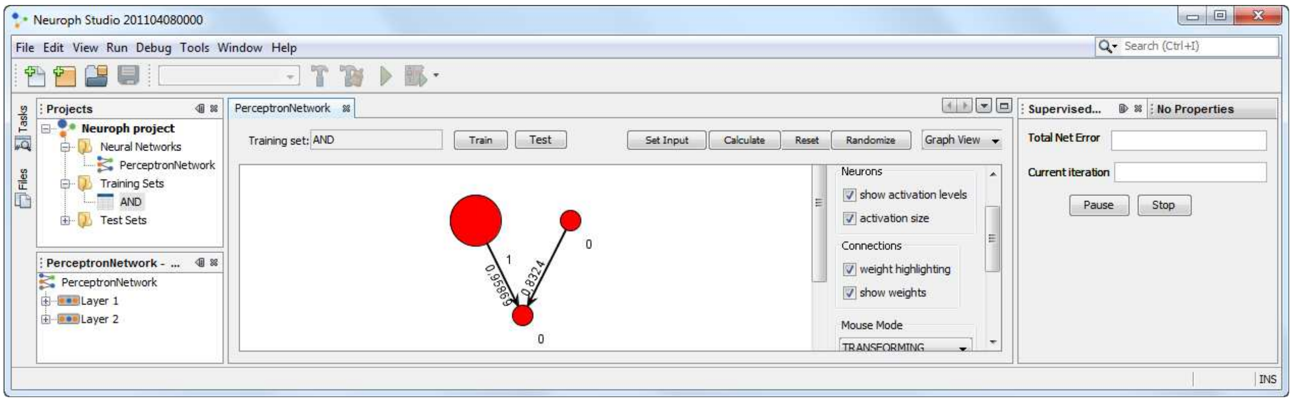
This will open Set Network Input dialog in which you can enter input values for network delimited with space.



The result of network test is shown on picture below. Network learned logical AND function. As we can see the output neuron has value 0. Test the network to see how it behaves for other input values.



You can select graph view to see different network view.



6. Creating Neural Networks 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);

// create training set
DataSet trainingSet =
                     new  DataSet(2, 1);
// add training data to training set (logical OR function)
trainingSet. addRow (new DataSetRow (new double[]{0, 0},
                                      new double[]{0}));
trainingSet. addRow (new DataSetRow (new double[]{0, 1},
                                      new double[]{1}));
trainingSet. addRow (new DataSetRow (new double[]{1, 0},
                                      new double[]{1}));
trainingSet. addRow (new DataSetRow (new double[]{1, 1},
                                      new double[]{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");

// set network input
neuralNetwork.setInput(1, 1);

// calculate network
neuralNetwork.calculate();

// get network output
double[] networkOutput = neuralNetwork.getOutput();
```

This example show the basic usage of neural network created with *Neuroph*.

To be able to use this in your programs, you must provide a reference to Neuroph Library jar file *neuroph-core-xx.jar* in your project (in NetBeans right click project, then Properties > Libraries > Add JAR/Folder, and choose jars). Also you must import the corresponding classes/packages, like org.neuroph.core, *org.neuroph.core.learning* and *org.neuroph.nnet.*

For more examples see source code in ***org.neuroph.samples*** package (which is located under Samples Maven project module) and **Help in NeurophStudio**.

List of all supported NN architectures, is available in *Neuroph* API documentation (see ***org.neuroph.nnet*** package).

7. Web Links

http://neuroph.sourceforge.net Official Neuroph site
http://en.wikipedia.org/wiki/Neuroph Neuroph on Wikipedia
http://www.oracle.com/technetwork/articles/java/nbneural-317387.html OTN article
Neuroph: Smart Java Apps with Neural Networks 1 (Part 1, interview on NetBeans Zone)
Neuroph: Smart Java Apps with Neural Networks 2 (Part 2, interview on NetBeans Zone)
Neuroph: Smart Java Apps with Neural Networks 3 (Part 3, interview on NetBeans Zone)
Building Smart Java Applications with Neural Networks, Using the Neuroph Framework, JavaOne2012 session video
Neuroph application samples 30 application samples for misc data sets

For more usefull links see: http://neuroph.sourceforge.net/links.html