POLITECNICO DI MILANO
Department of Electronics and Information (DEI)

poliGMDH (NeuralNetwork v. 4)
User Guide

1. Installation and execution of the Java development environment program and of poliGMDH.

Java(TM) 2 SDK, Standard Edition is a development environment for building java applications, applets, and/or components. The version of the Java machine to run the program is 5.

The poliGMDH application has been checked with Windows XP, MAC OS X, Linux..
A minimum availability of 300Mb RAM is necessary.

[1] The Java 2 SDK, Standard Edition, is a product of Sun Microsystems(TM), Inc. This product includes code licensed from RSA Security. Copyright 2002 Sun Microsystems, Inc., 901 San Antonio Road, Palo Alto, California 94303, U.S.A. All rights reserved.

2. functions in poliGMDH.

2.1. Why Use poliGMDH.

The present implementation of poliGMDH builds a Neural Network starting from a training set given in XML format. The net is saved in XML format for future recalling. There is a also a validation step based on the computation of R2 on an external test set.

Different uses of the system are possible in model construction and data mining. Any user has the ability of stand-alone modeling and information management via the toolbox apart of the programs graphical user interface (GUI).

GMDH is the realization of an inductive approach for mathematical modeling of complex systems. Basically it incrementally builds a multilayer network, beginning with the combination of the input layer neurons. This generates an internal layer and then it 'prunes' all neurons, selecting the best. New layers are continually added, 'pruning' them until a final usable) criterion is met.

Hence creating models in the program where the user can make estimations for single chemicals, retrieve the results of all the QSAR estimates for any models covering the appropriate chemical domain. The user can receive summary information on the validation results of the models used.

2.2. Building a network.

A new neural network is simply built by following the steps indicated herein.

As depicted in figure 1, the File menu has a list of options pertaining to a networks construction, saving and configuring in the XML standard.

There are three polynomials of choice to construct a network, before any data is passed.

- Ivakhnenko polynomial
- Bi-Quadratic polynomial
- Exponential polynomial

These polynomials available to the user define the type of networks which can be constructed. Modeling requires a system that yields a continuous answer by means of a polynomial from each input of the function.

To commence, click on the *File* tab and choose *New,* then choose the desired polynomial.

A GMDH network has neurons of two types: distribution neurons and executive neurons. Distribution neurons are only contained in input layer. They distribute components of the input vector to the neurons of first hidden layer. Executive neurons form all other layers. These neurons have primary influence on the behaviour of the network. Each neuron has exactly two inputs and one output, applied to the neurons of next layer. There are applied signals from outputs of previous layer to the inputs.

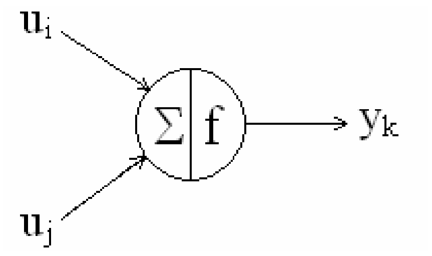


Figure 1: *General architecture of a neuron with two inputs*

Three polynomials are available to the user to define the type of input combination.

a -- The Ivakhnenko polynomial

This is expressed by the following equation:

$$y_1 = Ax_1x_2 + Bx_1^2 + Cx_2^2 + Dx_1 + Ex_2 + F$$

Every element in the network implements a non linear function of its input. One of them has two inputs (z_{ik}-1 & z_{jk}-1) and only one output (z_{mk}). The transfer function is a quadratic polynomial type in the form:

$z_{mk} = a_{mk} (z_{ik}-1)^2 + b_{mk} (z_{ik}-1)(z_{jk}-1) + c_{mk} (z_{jk}-1)^2 + d_{mk} z_{ik}-1 + e_{mk} z_{jk}-1 + f_{mk}$

The output can be expressed in an array function of the input x: $y = f(x)$

b --The Modified bi-quadratic polynomial

This is expressed by the following equation:

$$y_2 = Ax_1x_2 + Bx_1 + Cx_2 + D$$

It is derived from the Ivakhnenko polynomial in a simplified form:

$z_{mk} = a_{mk} (z_{ik}-1)(z_{jk}-1) + b_{mk} z_{ik}-1 + c_{mk} z_{jk}-1 + d_{mk}$

Where the single quadratic terms do not match, but the only second grade term is the mixed term related to evaluation, this function provides the best results in respect to the coefficient of determination.

c-- The Exponential function

This is expressed by the following equation:

$$y_3 = exp[-(y_2^2)]$$

This function is introduced as an innovative experimentation, presented as:

$zmk = e_{xp}\{-[a_{mk}\ (z_{ik}-1)(z_{jk}-1) + b_{mk}\ z_{ik}-1 + c_{mk}\ z_{jk}-1 + d_{mk}]^2\}$

The expression between the square parentheses is the bi-quadratic polynomial function already introduced.

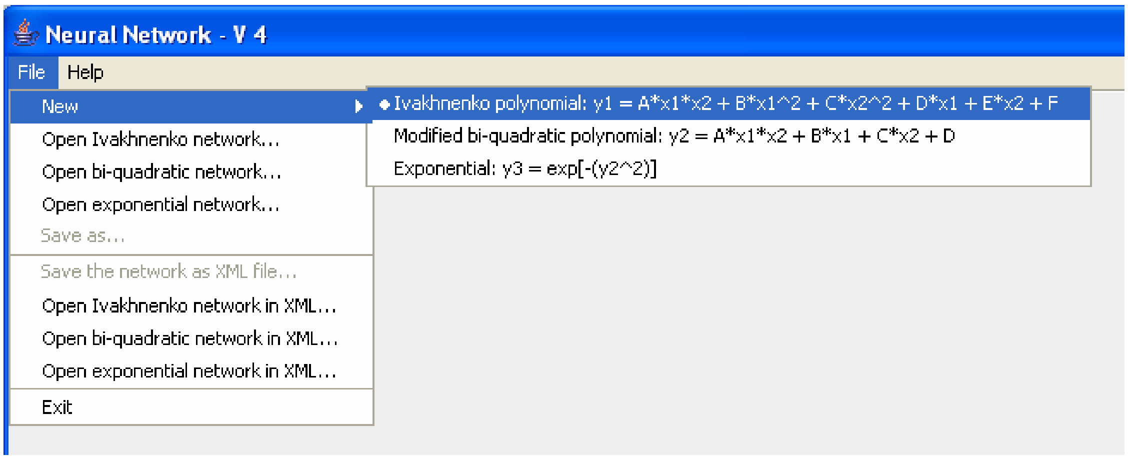
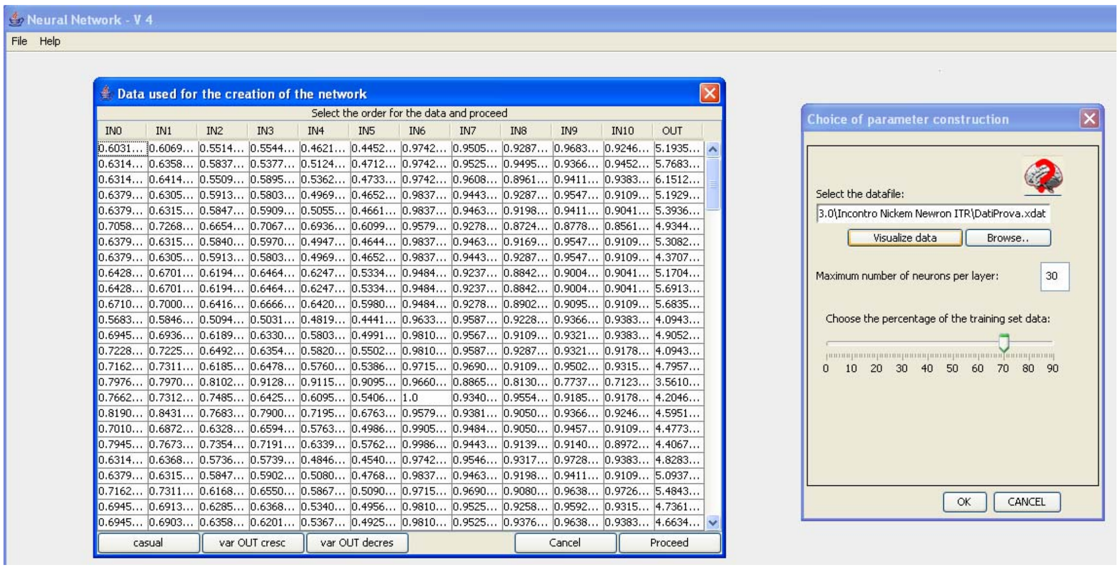

Neural Network - V 4
File
Help
New
Open Ivakhnenko network...
Open bi-quadratic network...
Open exponential network...
Save as...
Save the network as XML file...
Open Ivakhnenko network in XML...
Open bi-quadratic network in XML...
Open exponential network in XML...
Exit
Ivakhnenko polynomial: y1 = A*x1*x2 + B*x1^2 + C*x2^2 + D*x1 + E*x2 + F
Modified bi-quadratic polynomial: y2 = A*x1*x2 + B*x1 + C*x2 + D
Exponential: y3 = exp[-(y2^2)]

Figure 2: *To create a new neural network, example; an Ivakhnenko polynomial*

From here you must determine the output parameter for which the network is to be constructed. That is, for the predefined dataset (datafile with .xdat extension) which is a table of numbers, in which the first columns represent the inputs, and the last columns represent the outputs.

Figure 3 depicts where to type in the path of which the datafile is located, or alternatively click on “Browse...” to select from the files available on the computer.



IN0	IN1	IN2	IN3	IN4	IN5	IN6	IN7	IN8	IN9	IN10	OUT
0.6031...	0.6069...	0.5514...	0.5544...	0.4621...	0.4452...	0.9742...	0.9505...	0.9287...	0.9683...	0.9246...	5.1935...
0.6314...	0.6358...	0.5837...	0.5377...	0.5124...	0.4712...	0.9742...	0.9525...	0.9495...	0.9366...	0.9452...	5.7683...
0.6314...	0.6414...	0.5509...	0.5895...	0.5362...	0.4733...	0.9742...	0.9608...	0.8961...	0.9411...	0.9383...	6.1512...
0.6379...	0.6305...	0.5913...	0.5803...	0.4969...	0.4652...	0.9837...	0.9443...	0.9287...	0.9547...	0.9109...	5.1929...
0.6379...	0.6315...	0.5847...	0.5909...	0.5055...	0.4661...	0.9837...	0.9463...	0.9198...	0.9411...	0.9041...	5.3936...
0.7058...	0.7268...	0.6654...	0.7067...	0.6936...	0.6099...	0.9579...	0.9278...	0.8724...	0.8778...	0.8561...	4.9344...
0.6379...	0.6315...	0.5840...	0.5970...	0.4947...	0.4644...	0.9837...	0.9463...	0.9169...	0.9547...	0.9109...	5.3082...
0.6379...	0.6305...	0.5913...	0.5803...	0.4969...	0.4652...	0.9837...	0.9443...	0.9287...	0.9547...	0.9109...	4.3707...
0.6428...	0.6701...	0.6194...	0.6464...	0.6247...	0.5334...	0.9484...	0.9237...	0.8842...	0.9004...	0.9041...	5.1704...
0.6428...	0.6701...	0.6194...	0.6464...	0.6247...	0.5334...	0.9484...	0.9237...	0.8842...	0.9004...	0.9041...	5.6913...
0.6710...	0.7000...	0.6416...	0.6666...	0.6420...	0.5980...	0.9484...	0.9278...	0.8902...	0.9095...	0.9109...	5.6835...
0.5683...	0.5846...	0.5094...	0.5031...	0.4819...	0.4441...	0.9633...	0.9587...	0.9228...	0.9366...	0.9383...	4.0943...
0.6945...	0.6936...	0.6189...	0.6330...	0.5803...	0.4991...	0.9810...	0.9567...	0.9109...	0.9321...	0.9383...	4.9052...
0.7228...	0.7225...	0.6492...	0.6354...	0.5820...	0.5502...	0.9810...	0.9587...	0.9287...	0.9321...	0.9178...	4.0943...
0.7162...	0.7311...	0.6185...	0.6478...	0.5760...	0.5386...	0.9715...	0.9690...	0.9109...	0.9502...	0.9315...	4.7957...
0.7976...	0.7970...	0.8102...	0.9128...	0.9115...	0.9095...	0.9660...	0.8865...	0.8130...	0.7737...	0.7123...	3.5610...
0.7662...	0.7312...	0.7485...	0.6425...	0.6095...	0.5406...	1.0	0.9340...	0.9554...	0.9185...	0.9178...	4.2046...
0.8190...	0.8431...	0.7683...	0.7900...	0.7195...	0.6763...	0.9579...	0.9381...	0.9050...	0.9366...	0.9246...	4.5951...
0.7010...	0.6872...	0.6328...	0.6594...	0.5763...	0.4986...	0.9905...	0.9484...	0.9050...	0.9457...	0.9109...	4.4773...
0.7945...	0.7673...	0.7354...	0.7191...	0.6339...	0.5762...	0.9986...	0.9443...	0.9139...	0.9140...	0.8972...	4.4067...
0.6314...	0.6368...	0.5736...	0.5739...	0.4846...	0.4540...	0.9742...	0.9546...	0.9317...	0.9728...	0.9383...	4.8283...
0.6379...	0.6315...	0.5847...	0.5902...	0.5080...	0.4768...	0.9837...	0.9463...	0.9198...	0.9411...	0.9109...	5.0937...
0.7162...	0.7311...	0.6168...	0.6550...	0.5867...	0.5090...	0.9715...	0.9690...	0.9080...	0.9638...	0.9726...	5.4843...
0.6945...	0.6913...	0.6285...	0.6368...	0.5340...	0.4956...	0.9810...	0.9525...	0.9258...	0.9592...	0.9315...	4.7361...
0.6945...	0.6903...	0.6358...	0.6201...	0.5367...	0.4925...	0.9810...	0.9525...	0.9376...	0.9638...	0.9383...	4.6634...

Select the datafile: 3.0\Incontro Nickem Newron ITR\DatiProva.xdat

Maximum number of neurons per layer: 30

Choose the percentage of the training set data:

Figure 3: *Possibility of viewing and altering data of network*

An option to choose the maximum number of neurons per layer is available. By restricting the number of neurons, this can possibly reduce the accuracy of the network.

To select the desired datafile, click on “loading data” then you will be prompted to normalize this datafile. To normalize a datafile, is to scale and/or transform it. In most cases, it should be the nature of the problem rather than the method of solving it that dictates the criteria for normalizing input variables).

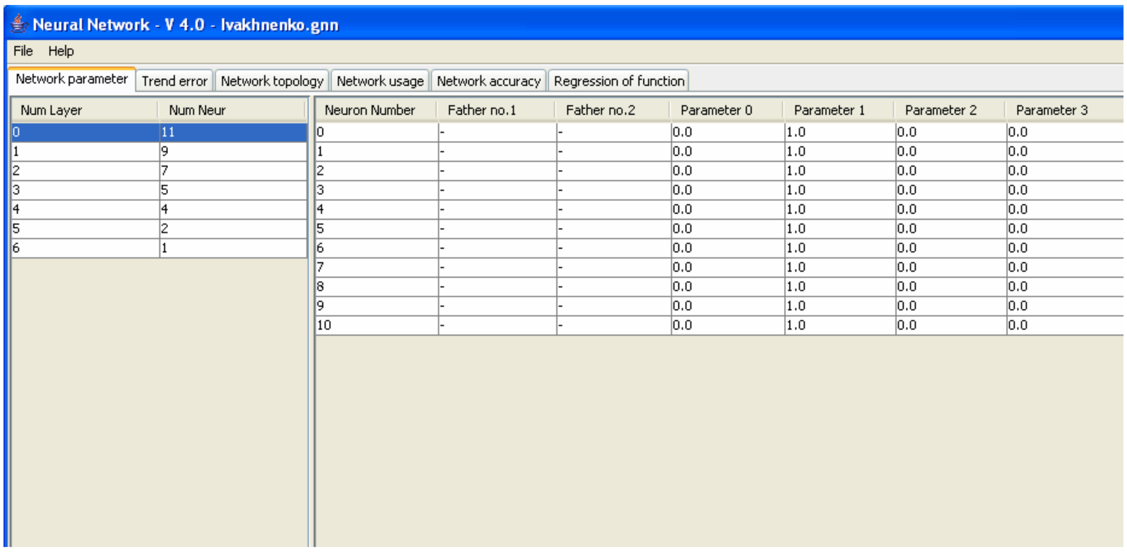
Secondly decide whether you would like a logarithmic output. A logarithmic output will allow for a logarithmic graph to be depicted in the program.

The table containing the data (as in figure 3) will be displayed. To select the order, you must choose from the variation of the increasing outlay to the variation of the decreasing output. Finally, specify the name of the file with a .gnn extension in order to save the created network.

Once a network has been established the user has several options available from the GUI. Testing it is possible with combinations of other items and to also analyze the constituents of the particular network.

The following pages are available from the tabs at the top of the programs screen:

- Network parameter
- Trend error
- Network topology
- Network usage
- Network accuracy
- Regression of function



Num Layer	Num Neur
0	11
1	9
2	7
3	5
4	4
5	2
6	1

Neuron Number	Father no.1	Father no.2	Parameter 0	Parameter 1	Parameter 2	Parameter 3
0	-	-	0.0	1.0	0.0	0.0
1	-	-	0.0	1.0	0.0	0.0
2	-	-	0.0	1.0	0.0	0.0
3	-	-	0.0	1.0	0.0	0.0
4	-	-	0.0	1.0	0.0	0.0
5	-	-	0.0	1.0	0.0	0.0
6	-	-	0.0	1.0	0.0	0.0
7	-	-	0.0	1.0	0.0	0.0
8	-	-	0.0	1.0	0.0	0.0
9	-	-	0.0	1.0	0.0	0.0
10	-	-	0.0	1.0	0.0	0.0

Figure 4 *Networks layers constituents (parameters)*

As the network has been constructed, the initial of the six tabs is automatically displayed. Network Parameter is displayed in figure 4 which is separated in two sections. The left section has the lists for selecting between layer number and neuron number. This allows the user to select various layers and neurons (in layers) and eventually view the constituents of the layers (in the right section of the page). A network physically begins (on the left) with the “father no.1” then “father no.2” columns of neurons, then followed by ‘x’ number of parameters (columns). This is how the layout of the network parameters tab is described.

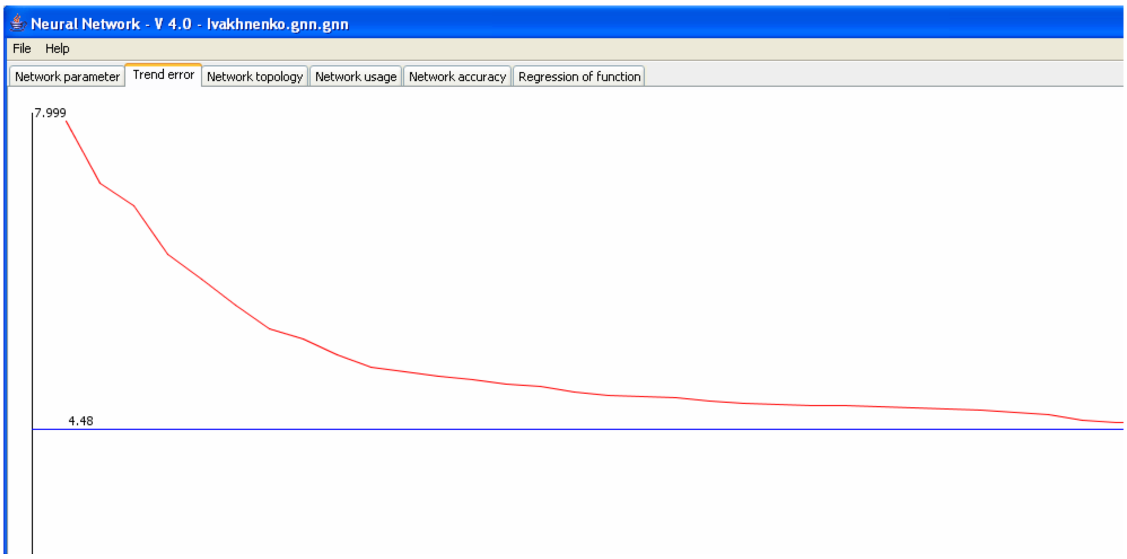


Figure 5: *Networks trend error*

The Trend error presented on the second tab (figure 5) depicts a graph of the trend of the network. This graphic of the MSE (visualization) is the system which is referenced when one begins the creation of a new network. When a network is being trained, it continues to do this until the MSE curve increases too much.

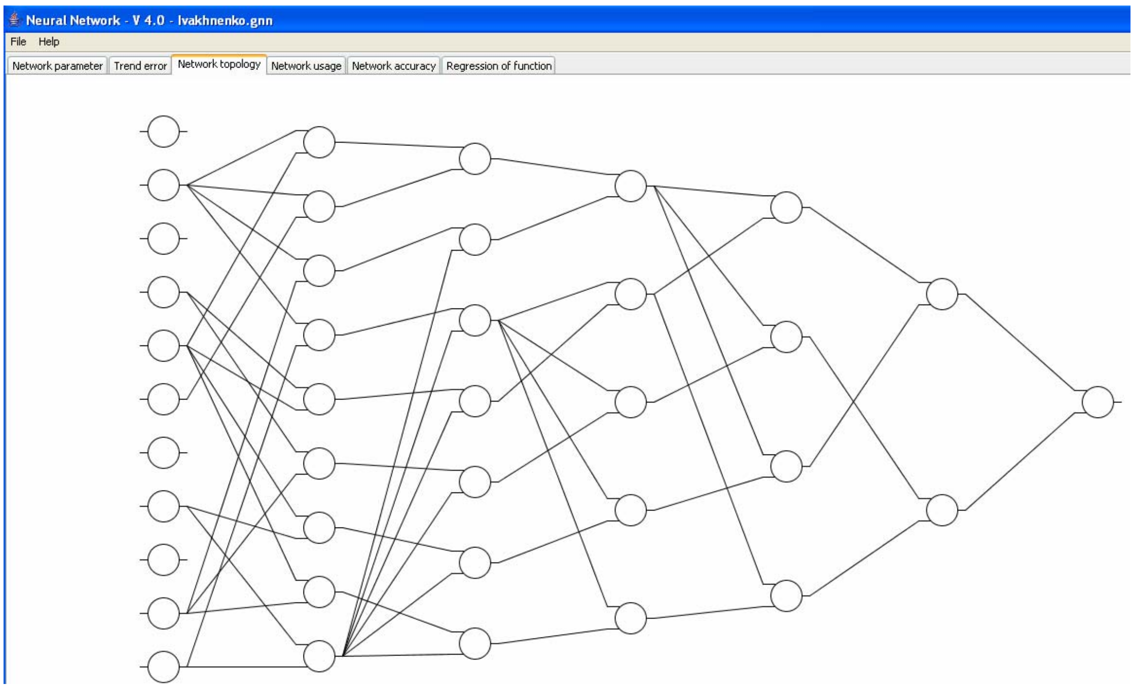
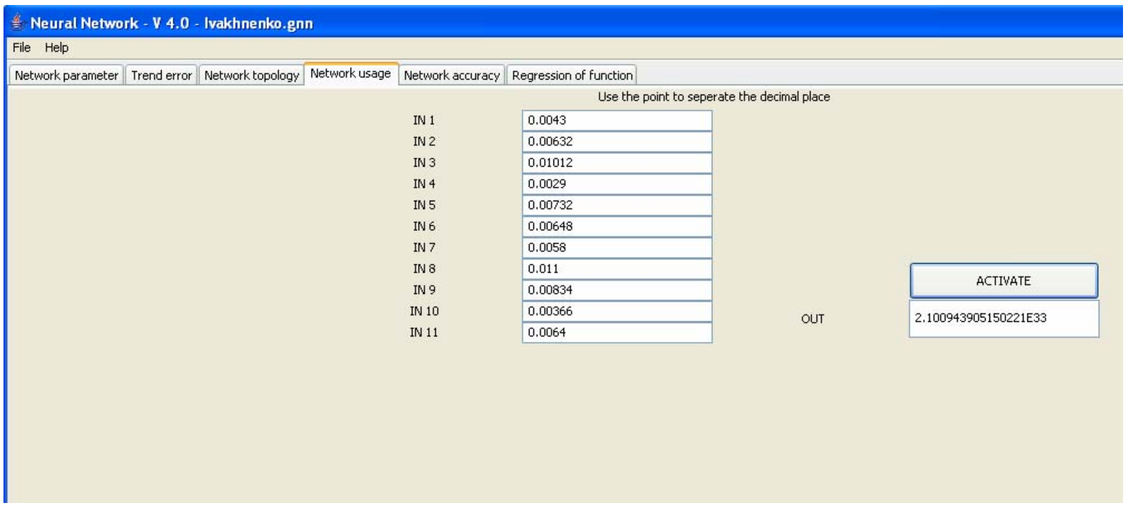


Figure 6: *One of the six pages to view or alter the network: here we use the net topology*

A networks topological structure can be viewed from the third tab (figure 6), which illustrates the general structure of the obtained network.



Use the point to seperate the decimal place

IN 1	0.0043
IN 2	0.00632
IN 3	0.01012
IN 4	0.0029
IN 5	0.00732
IN 6	0.00648
IN 7	0.0058
IN 8	0.011
IN 9	0.00834
IN 10	0.00366
IN 11	0.0064

OUT 2.100943905150221E33

Figure 7: *Network usage with the number of inputs*

One can reference other inputs to a constructed network by means of the fourth tab, Network usage, which is shown in figure 7. Dependant on the current network being used determines the number of inputs which can be entered on this page and ultimately calculated via the “Activate” button adjacent the column of input values on the same page.

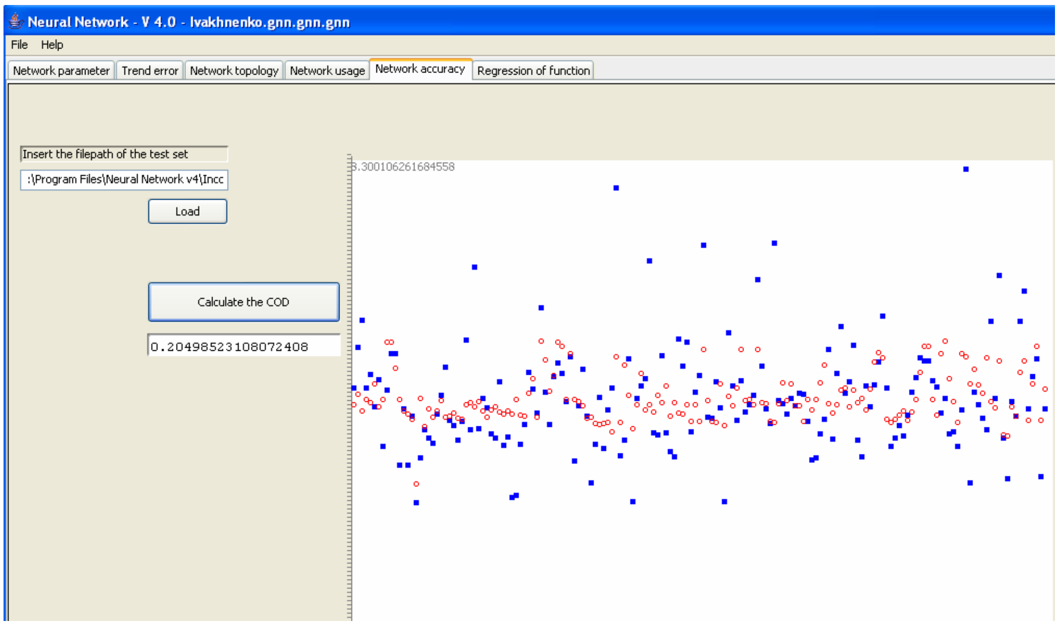


Figure 8: *Network accuracy by comparison with original dataset*

The creator of the network can make a comparison with the original dataset used for the current (possibly trained) network. By simply clicking on the load button of the fifth tab (network accuracy of figure 8) one can access the original .xdat dataset (or any other) which in turn graphically displays the calculations of the coefficient of determination against the current network, (the predicted (red) values against the real (blue) values).

The usage of the ‘COD parameter’ button ($R^{2)}$, the coefficient of determination represents the fraction of variability in y that can be explained by the variability in x. In other words, R^2 explains how much of the variability in the y can be explained by the fact that they are related to x, i.e., how close the points are to the regression line.

o If a model has perfect predictability, $R^2=1$.

o If a model has no predictive capability, $R^2=0$.

If the regression function is linear (yba = a + bx), the coefficient is the square of the linear correlation coefficient.

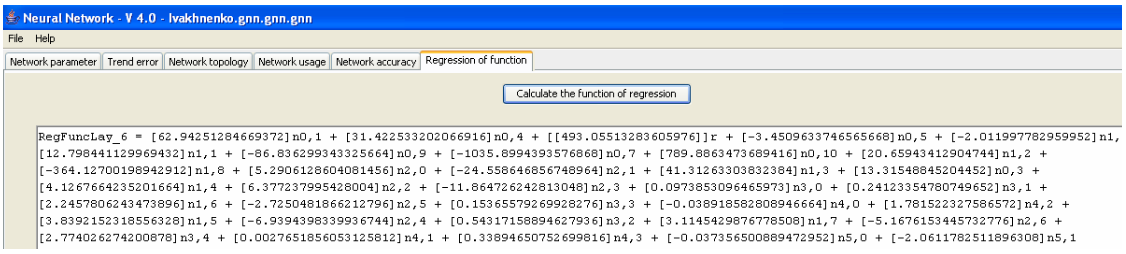
COD (R^2) can be expressed mathematically as:

$$r^2 = 1 - \frac{\Sigma (y_i - \hat{y}_i)^2}{\Sigma (y_i - \bar{y})^2}$$

where

y_i	observations
ybar	the mean of y
$\hat{y}$	residual, or what's left over after the regression has

explained away all the variability of the dependent variable.



```
RegFuncLay_6 = [62.94251284669372]n0,1 + [31.422533202066916]n0,4 + [[493.05513283605976]]r + [-3.4509633746565668]n0,5 + [-2.011997782959952]n1,
[12.798441129969432]n1,1 + [-86.836299343325664]n0,9 + [-1035.8994393576868]n0,7 + [789.8863473689416]n0,10 + [20.65943412904744]n1,2 +
[-364.12700198942912]n1,8 + [5.2906128604081456]n2,0 + [-24.558646856748964]n2,1 + [41.31263303832384]n1,3 + [13.31548845204452]n0,3 +
[4.1267664235201664]n1,4 + [6.377237995428004]n2,2 + [-11.864726242813048]n2,3 + [0.0973853096465973]n3,0 + [0.24123354780749652]n3,1 +
[2.2457806243473896]n1,6 + [-2.7250481866212796]n2,5 + [0.15365579269928276]n3,3 + [-0.038918582808946664]n4,0 + [1.781522327586572]n4,2 +
[3.8392152318556328]n1,5 + [-6.9394398339936744]n2,4 + [0.5431715889462793 6]n3,2 + [3.1145429876778508]n1,7 + [-5.1676153445732776]n2,6 +
[2.774026274200878]n3,4 + [0.0027651856053125812]n4,1 + [0.33894650752699816]n4,3 + [-0.037356500889472952]n5,0 + [-2.0611782511896308]n5,1
```

Figure 9: *Calculation of "Regression Function"*

When building a regression function to approximate a set of observations, R^2 or COD explains how much of the variability in the y can be explained by the fact that they are related to x, i.e., how close the points are to the regression line. The regression function can be calculated and hence visualized in the last tab of the program, regression of function depicted in figure 9 above. In the case of a comprehensive network, some time is required for the program to do the calculations of the regression function upon the usage of the button "Calculate the function of regression".

Once the net is considered good, the user can save it in XML format. The net is ready for recalling on new data.

To open a network (model recall) one must use the File menu as per new models, but initially you must specify the type of network desired.A trained network differs from untrained networks. As cited earlier, training a model requires normalization and the like. Such as the creation of a network, model building with the possibility to select different transfer functions must have occurred. Then a training set could have been expressed in XML format, with or without normalization. One must be aware of this when opening a model for any intention of manipulating the pre existing network.