ANN in JustNN
how to…

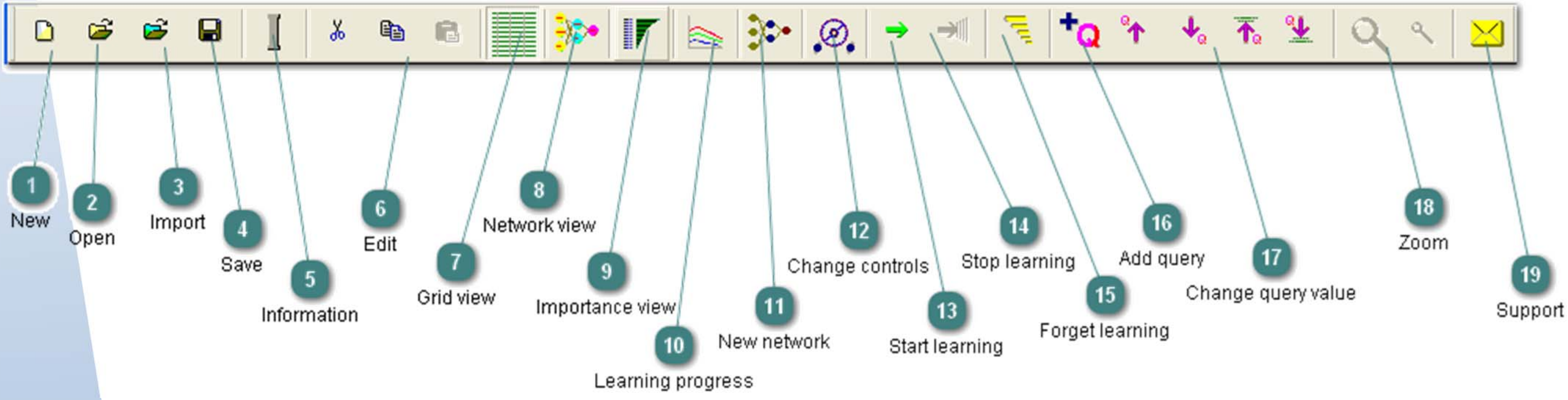
Prof. Raffaella Folgieri

JustNN

- Tool to create Artificial Neural Networks.
  - No coding, no programming languages.
  - Flexible, powerful, allows queries
  - Help in optimizing the net and choosing the right number of nodes
  - Possibility to impot data (txt, xls)
  - Download here: http://justnn.com/
  - Also a «plus» version (low cost for students): http://www.easynn.com/

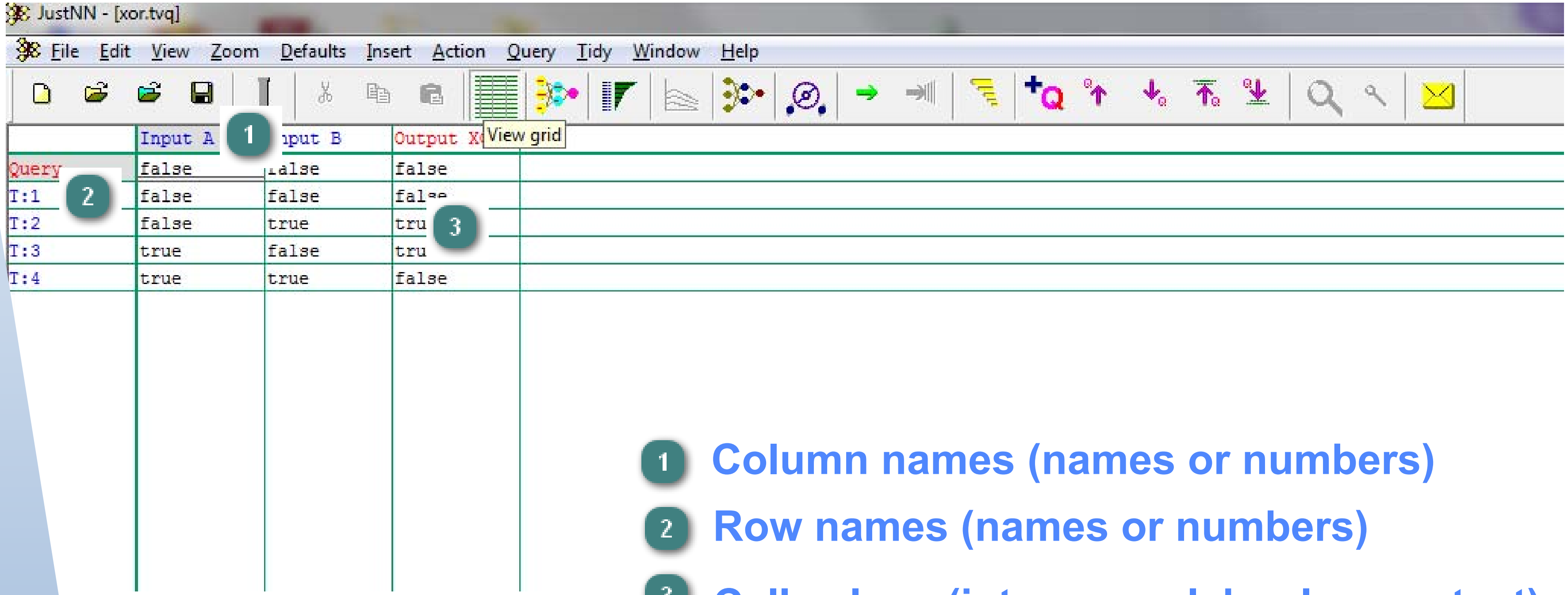
Workspace

- Buttons Bar. It shows the same cmmands present in the menu.



Workspace

- Data sheet

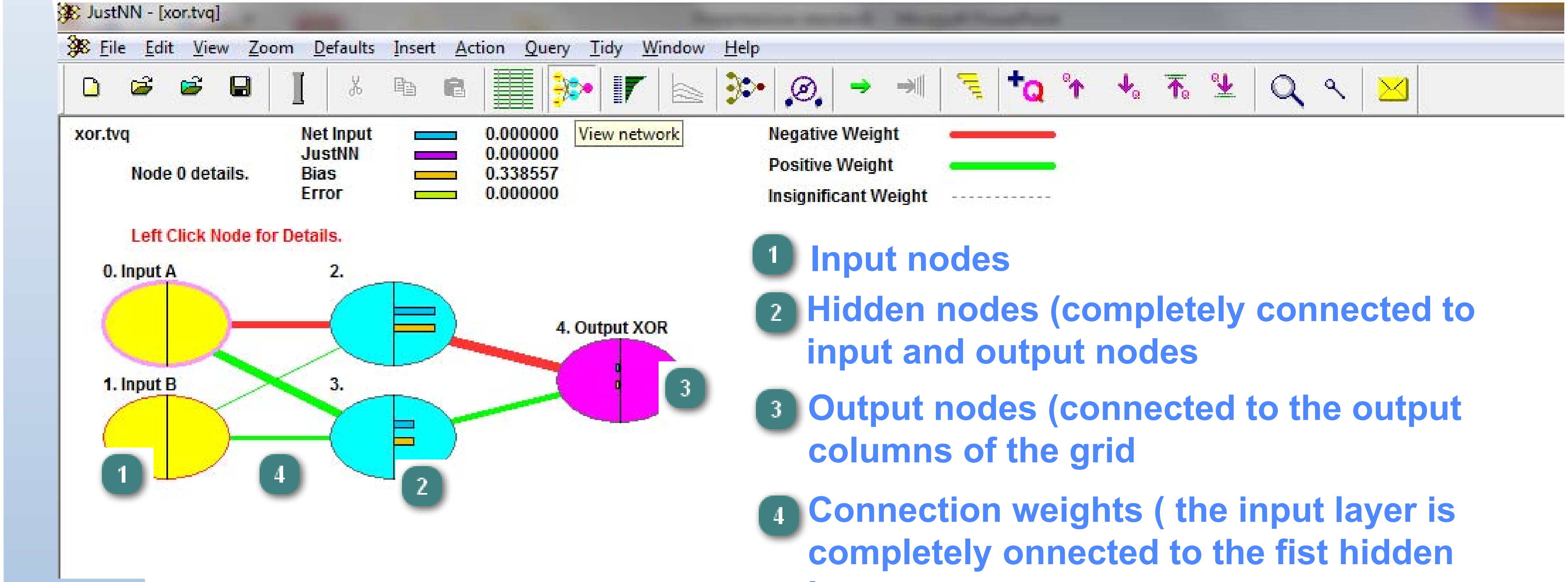

JustNN - [xor.tvq]
File Edit View Zoom Defaults Insert Action Query Tidy Window Help
View grid
1
2
3

	Input A	Input B	Output X
Query	false	false	false
T:1	false	false	false
T:2	false	true	true
T:3	true	false	true
T:4	true	true	false

1. Column names (names or numbers)
2. Row names (names or numbers)
3. Cell values (integer, real, boolean or text)

Workspace

- Network view



xor.tvq

Node 0 details.

Net Input	🟦	0.000000
JustNN	🟪	0.000000
Bias	🟧	0.338557
Error	🟨	0.000000

Negative Weight	🟥
Positive Weight	🟩
Insignificant Weight	- - - - - - - - - - -

Left Click Node for Details.

1 Input nodes

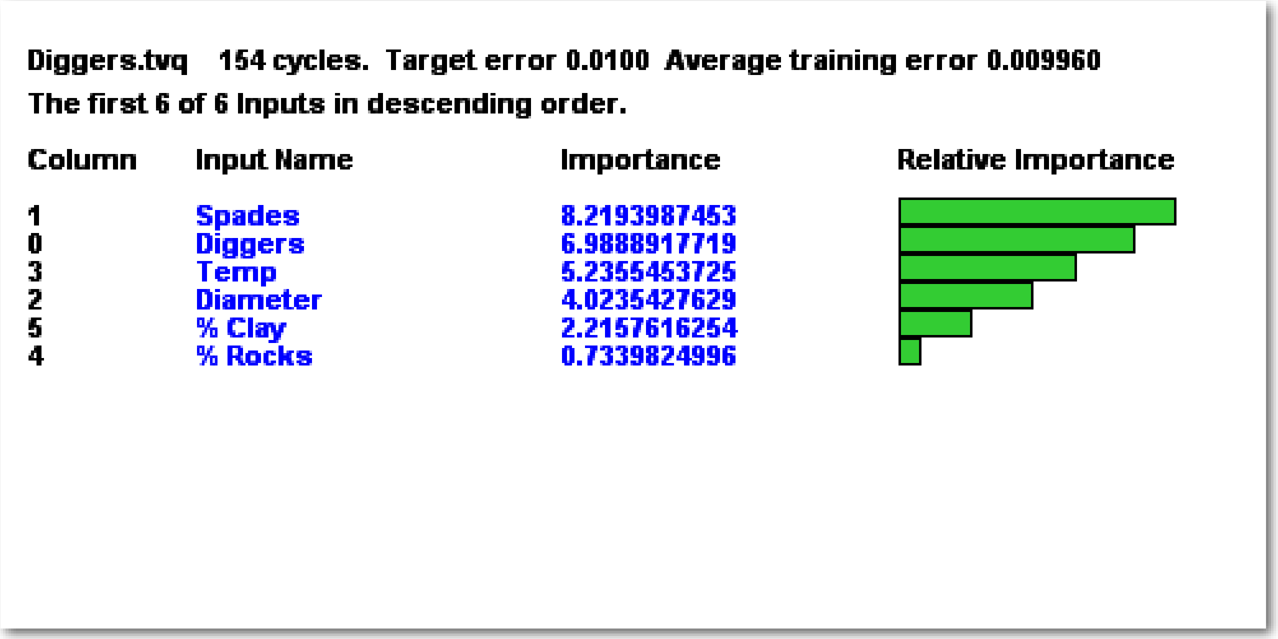
2 Hidden nodes (completely connected to input and output nodes

3 Output nodes (connected to the output columns of the grid

4 Connection weights (the input layer is completely onnected to the fist hidden layer.
Each connection has a weight updated when the net is learning (during the training phase)
The hidden layer are completely connected to the nodes of the following hidden layer or to the input layer

Workspace

- **Input Importance** shows the importance of each input column (that is the sum of the absolute weight of the connection from the input node to all the nodes of the first hidden layer). The inputs are shown in descent order of importance



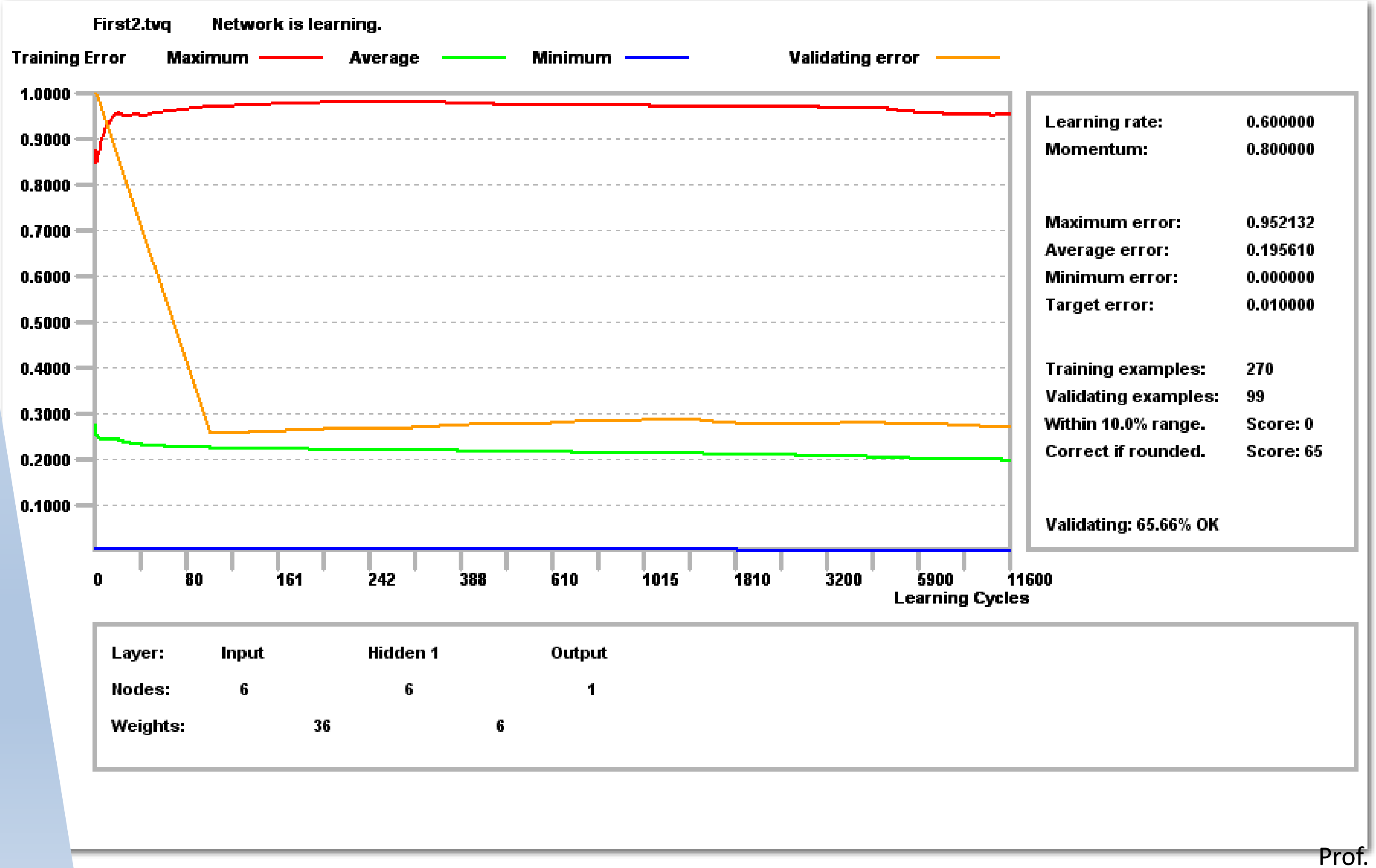
Diggers.tvq   154 cycles. Target error 0.0100 Average training error 0.009960

The first 6 of 6 Inputs in descending order.

Column	Input Name	Importance	Relative Importance
1	Spades	8.2193987453	
0	Diggers	6.9888917719	
3	Temp	5.2355453725	
2	Diameter	4.0235427629	
5	% Clay	2.2157616254	
4	% Rocks	0.7339824996	

Workspace

- **Learning Progress** show the progress of the net during the training (learning curve). The red line refers to the maximum error, the blu to the minimum error, the green represents the average. The orange line is the average of the validation error.



First2.tvq Network is learning.

Training Error Maximum 🟥 Average 🟩 Minimum 🟦 Validating error 🟧

Learning rate: 0.600000
Momentum: 0.800000

Maximum error: 0.952132
Average error: 0.195610
Minimum error: 0.000000
Target error: 0.010000

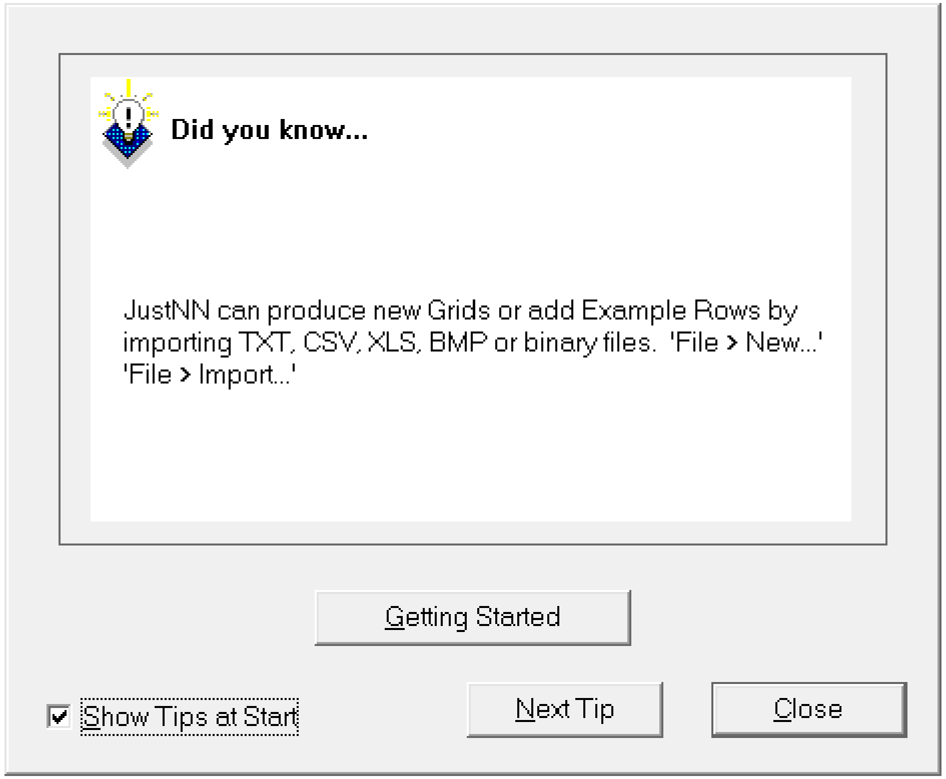
Training examples: 270
Validating examples: 99
Within 10.0% range. Score: 0
Correct if rounded. Score: 65

Validating: 65.66% OK

Layer:	Input		Hidden 1		Output
Nodes:	6		6		1
Weights:		36		6	

Example 1

- We will see an **ANN for the XOR problem**, following examples and data at disposition in JustNN folder.
- Run JustNN and follow the proposed steps in GETTING START from the dialog «Tips»



Did you know...

JustNN can produce new Grids or add Example Rows by importing TXT, CSV, XLS, BMP or binary files. 'File > New...' 'File > Import...'

Example 1

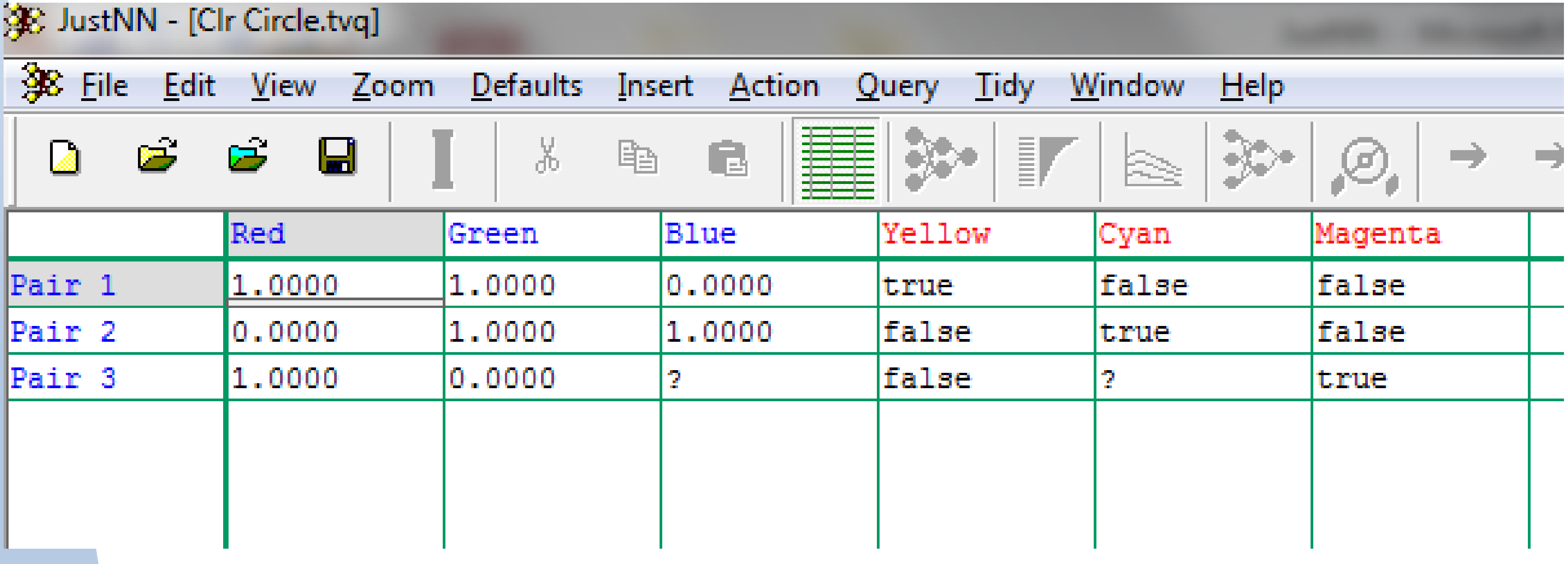
- Remember that the logical operator XOR (eXclusive OR) returns TRUE if **only one of the input** is TRUE. If the Input are both TRUE, the output is FALSE.
- Open the file **xor.tvq** in the folder **\JustNN\Samples**. The file already contains all the data we need.
- Columns and rows can be selected with a mouse click on the title cell.
- The input value (the row QUERY) can be changed from true to false or viceversa, using buttons 'Query > Increase‘ and 'Query > Max' o 'Query > Decrease' e 'Query > Min' in the toolbar. The output does not change because the net has not been trained yet

Example 1

- **To start training, select Action > Start Learning** or click on the green arrow on the button bar
- In this case the training will be very fast.
- Now try to change the input values (on the row QUERY) and you'll see that the results change (because the net has been trained)

Example 2

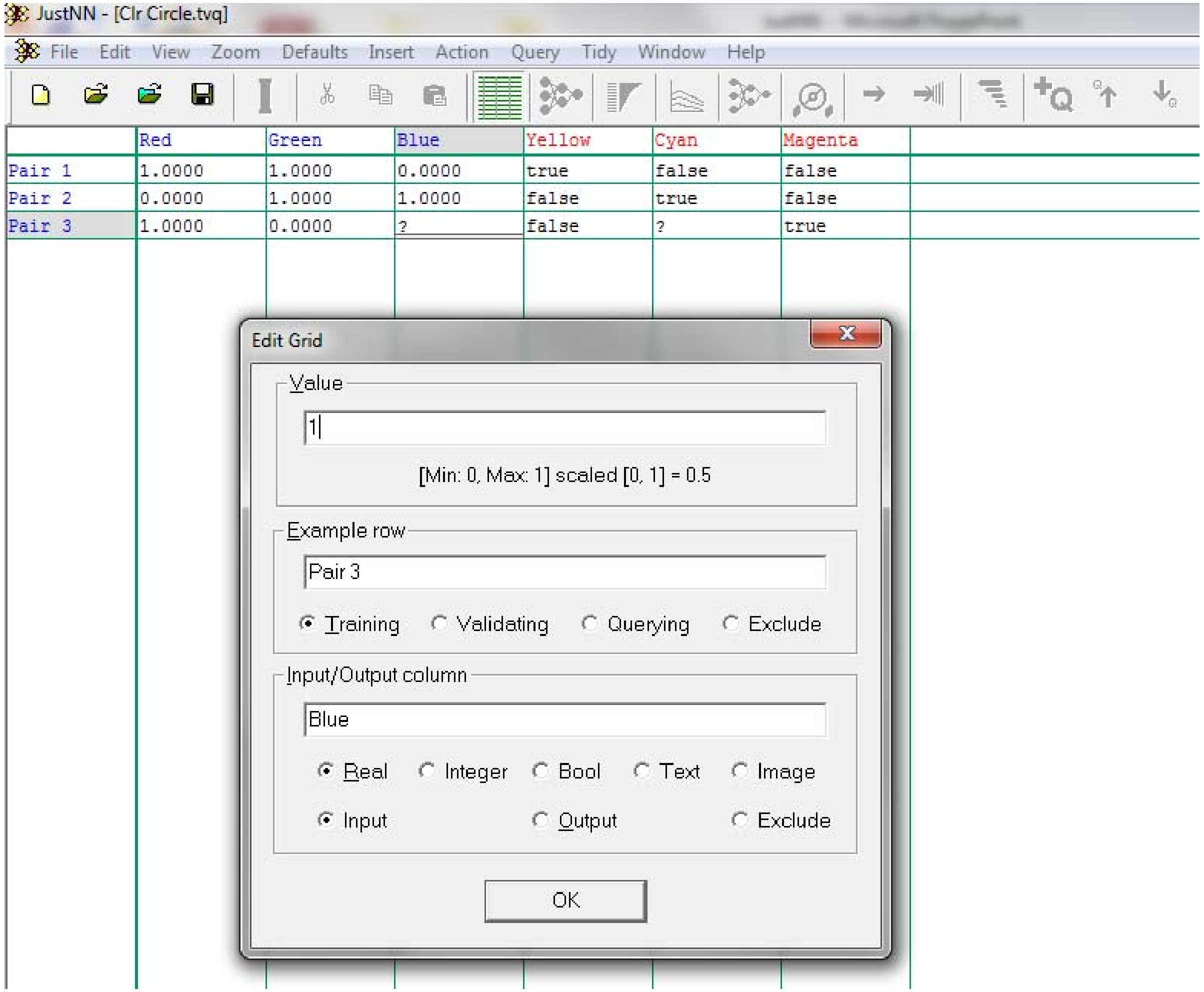
- We'll learn how insert data in a grid.
- Open the file **Clr Circle.tvq** in the folder **\JustNN\Samples**.
- Some data are already present, but you need to insert values (where you see the question mark).


JustNN - [Clr Circle.tvq]
File Edit View Zoom Defaults Insert Action Query Tidy Window Help

	Red	Green	Blue	Yellow	Cyan	Magenta
Pair 1	1.0000	1.0000	0.0000	true	false	false
Pair 2	0.0000	1.0000	1.0000	false	true	false
Pair 3	1.0000	0.0000	?	false	?	true

Example 2

- Double click on the cell between **Pair3** and **Blue** (or single click and write **1**) You will see a dialog box with the value. Click OK.
- Do the same with the cell between **Pair3** and **Cyan**. In this case we just need to write **f**.
- We completed the grid


JustNN - [Clr Circle.tvq]
File Edit View Zoom Defaults Insert Action Query Tidy Window Help

	Red	Green	Blue	Yellow	Cyan	Magenta
Pair 1	1.0000	1.0000	0.0000	true	false	false
Pair 2	0.0000	1.0000	1.0000	false	true	false
Pair 3	1.0000	0.0000	?	false	?	true

Edit Grid

Value

1

[Min: 0, Max: 1] scaled [0, 1] = 0.5

Example row

Pair 3

● Training ○ Validating ○ Querying ○ Exclude

Input/Output column

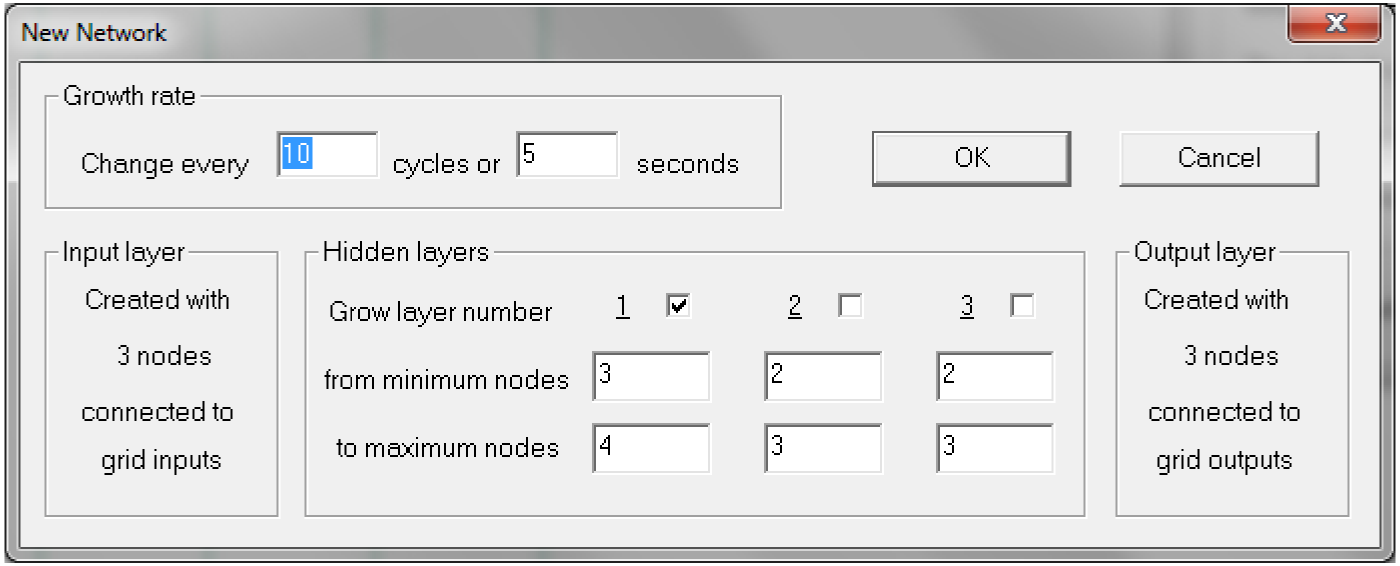
Blue

● Real ○ Integer ○ Bool ○ Text ○ Image

● Input ○ Output ○ Exclude

Example 2

- Select **menu Action > New Network** . In the dialog you will see all the paramenters correctly set by JustNN, automatically
- Click OK.



New Network

Growth rate

Change every 10 cycles or 5 seconds

Input layer

Created with

3 nodes

connected to

grid inputs

Hidden layers

Grow layer number 1 ☑ 2 □ 3 □

from minimum nodes 3 2 2

to maximum nodes 4 3 3

Output layer

Created with

3 nodes

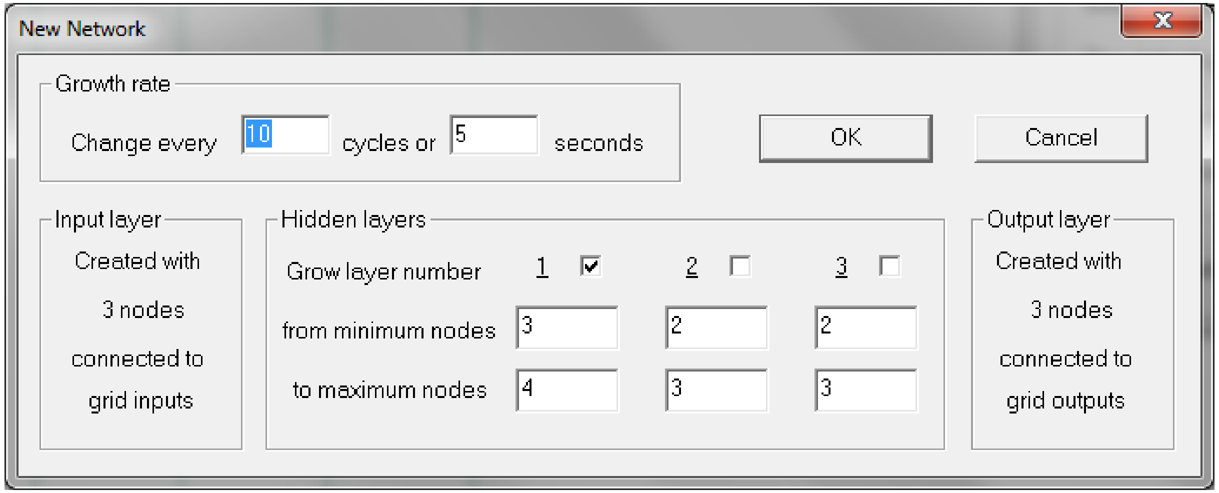
connected to

grid outputs

- Anyway, the ANN could need a lot of time to learn from data and could give few results in the test phase.

Example 2

- You can obtain a better ANN selecting the value **1** as **grow layer number**, allowing, in this way, to justNN to automatically detect the optimum number of nodes and connections. Only in really few cases, in fact, we could need more than one level of hidden nodes, but JustNN (selecting the values 2 or 3) can generate more levels.
- Every time the period of cicles or seconds ends, JustNN generates a new neural network different from the previous one, saving the best, at the end of the process.



New Network

Growth rate

Change every 10 cycles or 5 seconds

Input layer

Created with 3 nodes connected to grid inputs

Hidden layers

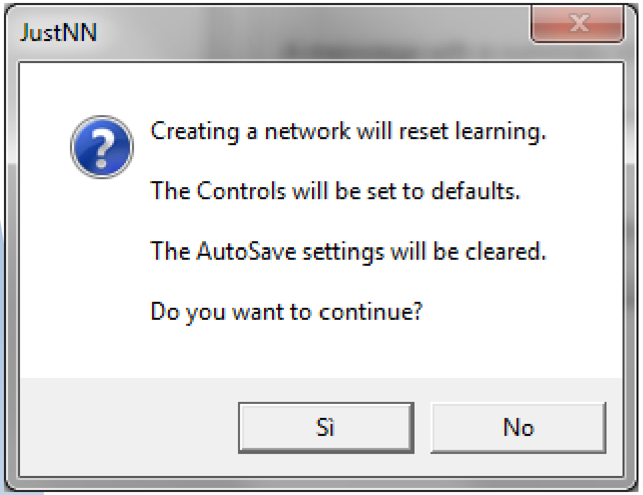
Grow layer number	1 ☑	2 □	3 □
from minimum nodes	3	2	2
to maximum nodes	4	3	3

Output layer

Created with 3 nodes connected to grid outputs

Example 2

- The dialog New Network allows to create an optimal net for our grid of values. Each net is trained for a short period and at the end JustNN selects the net producing the lowest error during the validation phase (the validation of the training).
- When you click Ok, you will see a message with a summary of the net's characteristics.
- Click "YES" in the following dialogs.

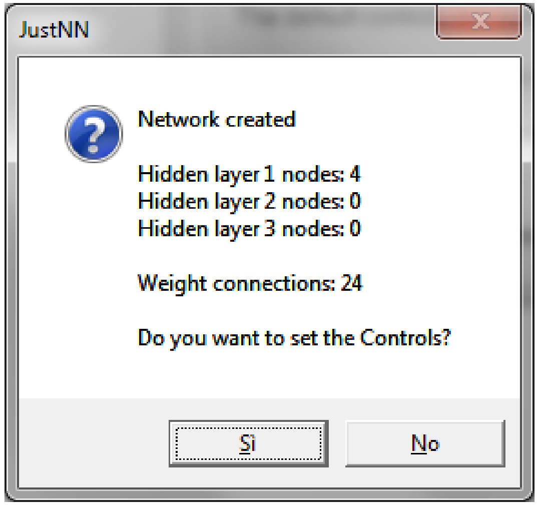


Creating a network will reset learning.

The Controls will be set to defaults.

The AutoSave settings will be cleared.

Do you want to continue?



Network created

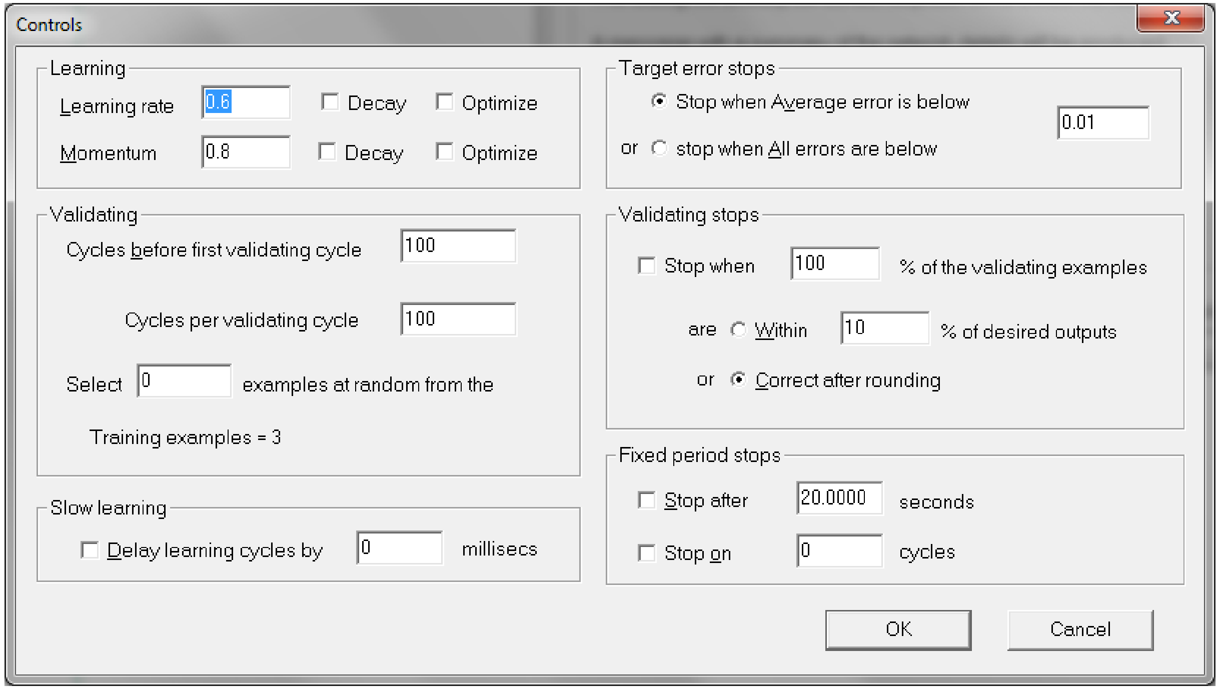
Hidden layer 1 nodes: 4
Hidden layer 2 nodes: 0
Hidden layer 3 nodes: 0

Weight connections: 24

Do you want to set the Controls?

Example 2

- When you create a NN, JustNN set the values to obtain the best net from data. All the controls allowing these setting are in the **Control dialog**.
- The **Learning Rate is set to 0.6** and you can change it with every value in the interval going from 0.1 to 10. Low values will produce a slow learning , while values greater than 1.5 could cause oscillations of wrong results.



Controls

Learning
Learning rate 0.6 ☐ Decay ☐ Optimize
Momentum 0.8 ☐ Decay ☐ Optimize

Target error stops
● Stop when Average error is below 0.01
or ○ stop when All errors are below

Validating
Cycles before first validating cycle 100
Cycles per validating cycle 100
Select 0 examples at random from the
Training examples = 3

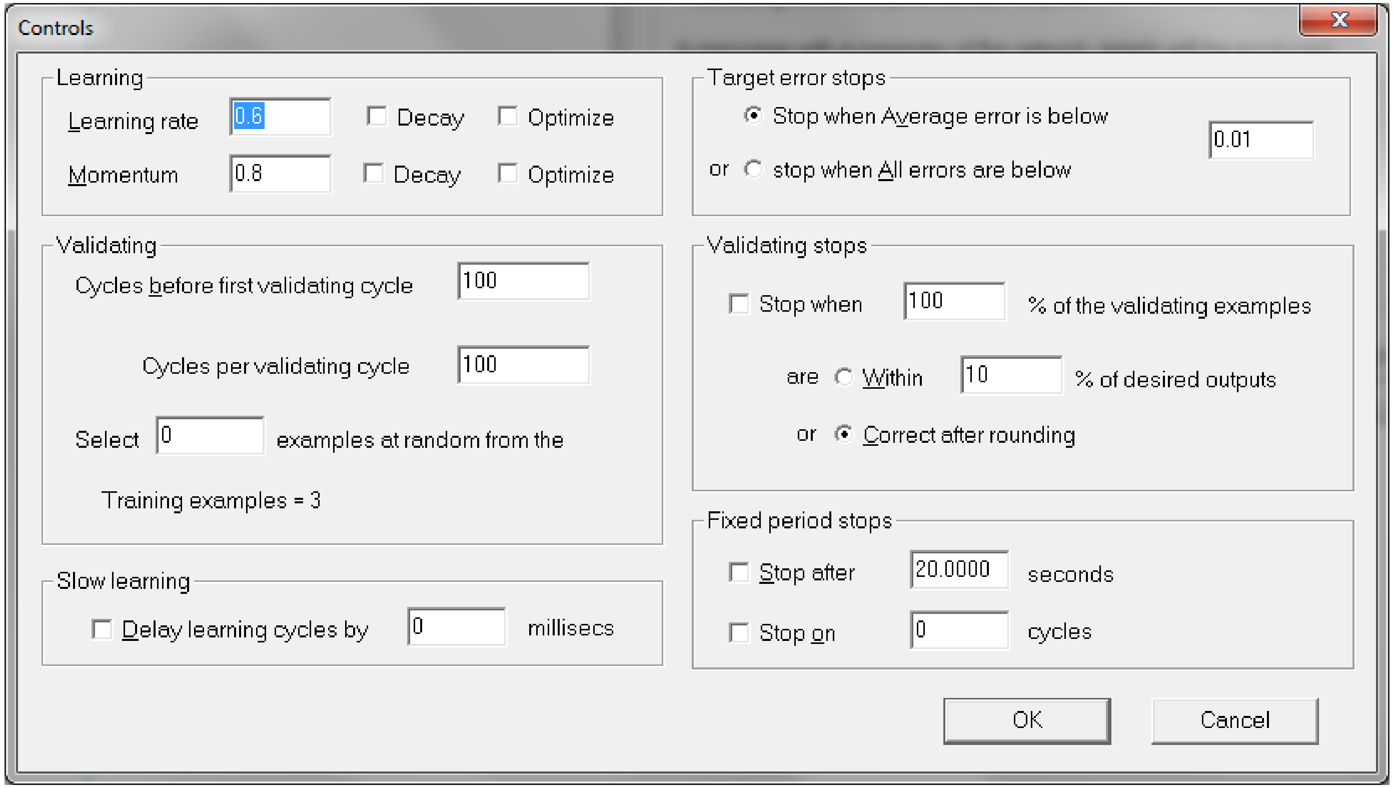
Validating stops
☐ Stop when 100 % of the validating examples
are ○ Within 10 % of desired outputs
or ● Correct after rounding

Slow learning
☐ Delay learning cycles by 0 millisecs

Fixed period stops
☐ Stop after 20.0000 seconds
☐ Stop on 0 cycles

Example 2

- If you click **Optimize**, JustNN will automatically set the right learning rate, trying several values for some cycle.
- Check **Decay** to automatically reduce the learning rate during the training if you have a wrong result or oscillations.



Controls

Learning

Learning rate 0.6 □ Decay □ Optimize

Momentum 0.8 □ Decay □ Optimize

Target error stops

● Stop when Average error is below 0.01

or ○ stop when All errors are below

Validating

Cycles before first validating cycle 100

Cycles per validating cycle 100

Select 0 examples at random from the

Training examples = 3

Validating stops

□ Stop when 100 % of the validating examples

are ○ Within 10 % of desired outputs

or ● Correct after rounding

Slow learning

□ Delay learning cycles by 0 millisecs

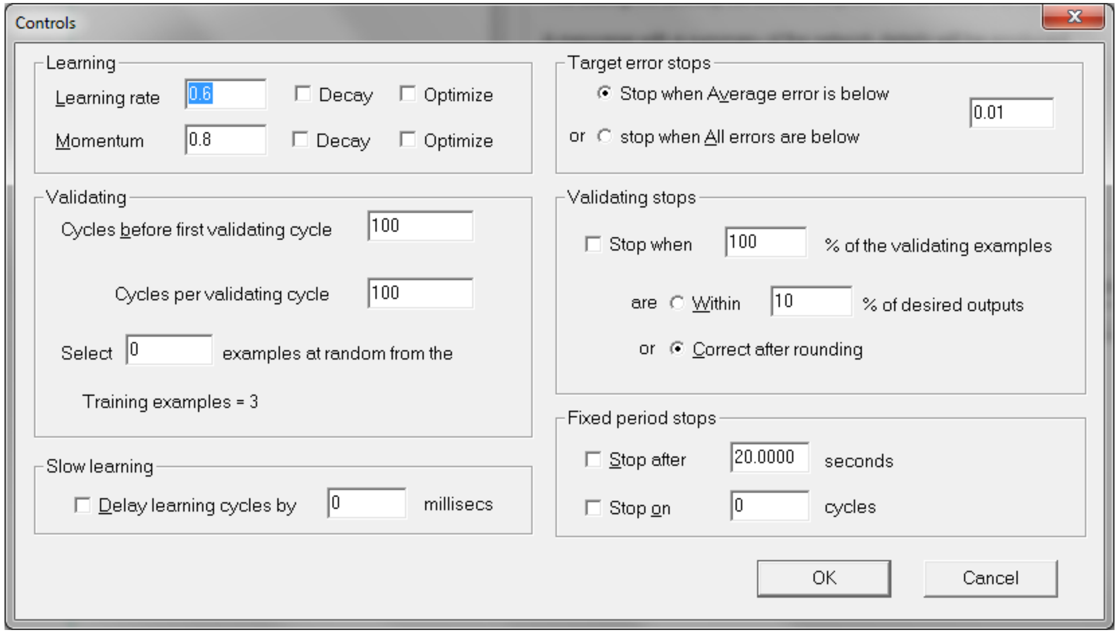
Fixed period stops

□ Stop after 20.0000 seconds

□ Stop on 0 cycles

Example 2

- **Momentum** is set to 0.8 and can be changed with a value from 0 to 0.9. Check **Optimize** to allow JustNN to automatically set the momentum. Check **Decay** if you want to automatically reduce the momentum during the learning if you have oscillations.
- The **Target Error** is set to 0.01 and can be changed with a value from 0 to 0.9 but values greater than 0.2 usually under-train the net. The learning process stops when the average of errors is lower than the target error.



Controls

Learning
Learning rate 0.6 □ Decay □ Optimize
Momentum 0.8 □ Decay □ Optimize

Target error stops
● Stop when Average error is below 0.01
or ○ stop when All errors are below

Validating
Cycles before first validating cycle 100
Cycles per validating cycle 100
Select 0 examples at random from the
Training examples = 3

Validating stops
□ Stop when 100 % of the validating examples
are ○ Within 10 % of desired outputs
or ● Correct after rounding

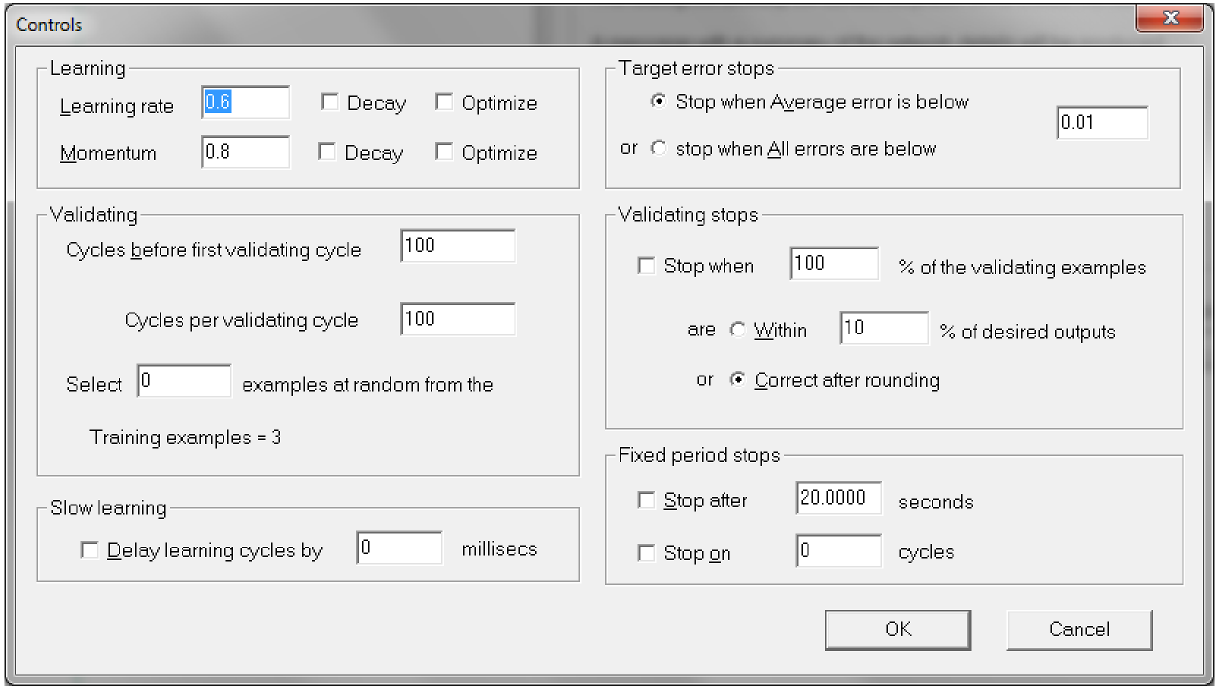
Slow learning
□ Delay learning cycles by 0 millisecs

Fixed period stops
□ Stop after 20.0000 seconds
□ Stop on 0 cycles

Prof. R. Folgieri

Example 2

- JustNN uses part of the data (in a random way) to train the net, and part for the validation process. You can decide to stop the learning when the target is achieved (ex. 100% or indicating a specific range – from 0 to 50%)
- The parameter **Slow learning** allows a slower learning (more accurate?)



Controls

Learning

Learning rate 0.6 □ Decay □ Optimize

Momentum 0.8 □ Decay □ Optimize

Target error stops

● Stop when Average error is below 0.01

or ○ stop when All errors are below

Validating

Cycles before first validating cycle 100

Cycles per validating cycle 100

Select 0 examples at random from the

Training examples = 3

Validating stops

□ Stop when 100 % of the validating examples

are ○ Within 10 % of desired outputs

or ● Correct after rounding

Slow learning

□ Delay learning cycles by 0 millisecs

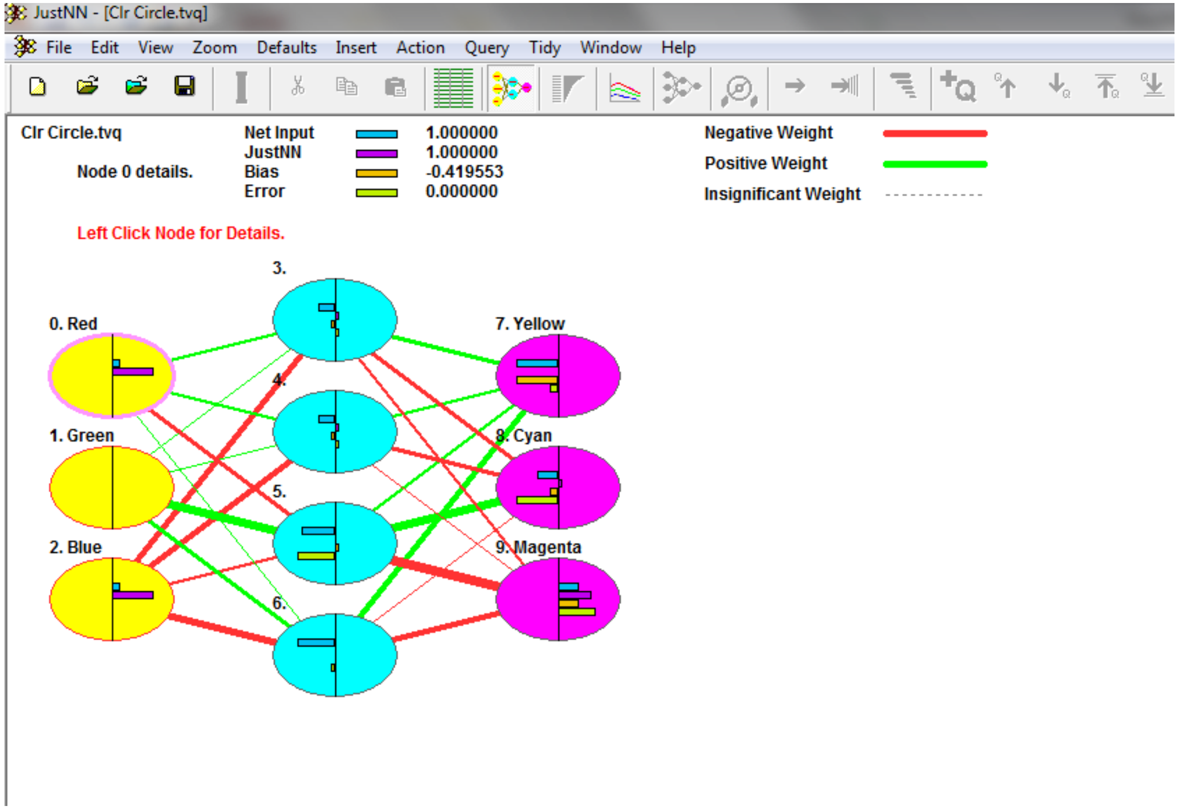
Fixed period stops

□ Stop after 20.0000 seconds

□ Stop on 0 cycles

Example 2

- The learning (traininng phase) is very fast and it automatically stops.
- Click on menu **View > Network** to see the ANN you have created.
- Click on **View>Grid** to visualize the grid.
- Now you can query the net from the grid.



Clr Circle.tvq

Node 0 details.

Net Input	🟦	1.000000
JustNN	🟪	1.000000
Bias	🟧	-0.419553
Error	🟨	0.000000

Negative Weight	🟥
Positive Weight	🟩
Insignificant Weight	------------

Left Click Node for Details.

Example 3

- We will treat now data related to horses races. Every row shows the characteristics of a horse as collected in different races. Following the same procedure seen in the previous examples, we will obtain a way to predict the result of a horse in a race.
- We will use the function IMPORT for our data

1. Chose **File > New** to create an empty grid
2. Now **File > Import** to importd data from **Races.txt** in the directory **\JustNN\Samples** (remember that you can import data also from excel, if you need).
3. In the dialog, select the delimitation among columns (ex. TAB)
4. Now indicata if the first words (as in our case) contain the name of the rows of the grid.
5. Click OK
6. In the second dialog box, click **Set names** to create the columns and go forward till the last one (in our case, we know that the last is the number 6)
7. When you are on the last column, remember to change the type) in **Output**
8. click OK.

Example 3

- Now menu **Action > New Network** to create the ANN
- Click OK in the first dialog (we already know it)
- Answer YES to the message “**Creating a network will reset learning**”
- Answer YES also to the following message
- In the control dialog:
  - Check **Optimize** either for **Learning Rate** and for **Momentum**
  - Put 200 in “**Select examples at random**” to create some validation examples (randomly selected by JustNN)
- Check “**Stop on cycle**” and put the value 1000. Then click OK
- Answer YES to optimize the controls and YES to start the learning
- Once done the learning phase, select the menu item **View > Information** to see the details of our network.
- Click on the **Refresh** button to be sure that the details are all shown.
- Look at the validation results: from 50 to 80% the “predicted” results are correct.
- Close the information dialog

Example 3

- Menu **Insert > Querying Example Row**
- If an '**Example Presets**' dialog appears, click OK to set the values to **unknown** (you will see '?‘ in the grid).
- Set the values **Runners** and **Distance** for a race.
- Select the query row with a double click where you see **Q:0** written in red
- Now chose from menu **Edit > Copy** and then **Edit > Paste** to create a row for each horse (for example 3). In WIN you will see the possible winners. Obviously, if you have only one horse, the result is more probable.

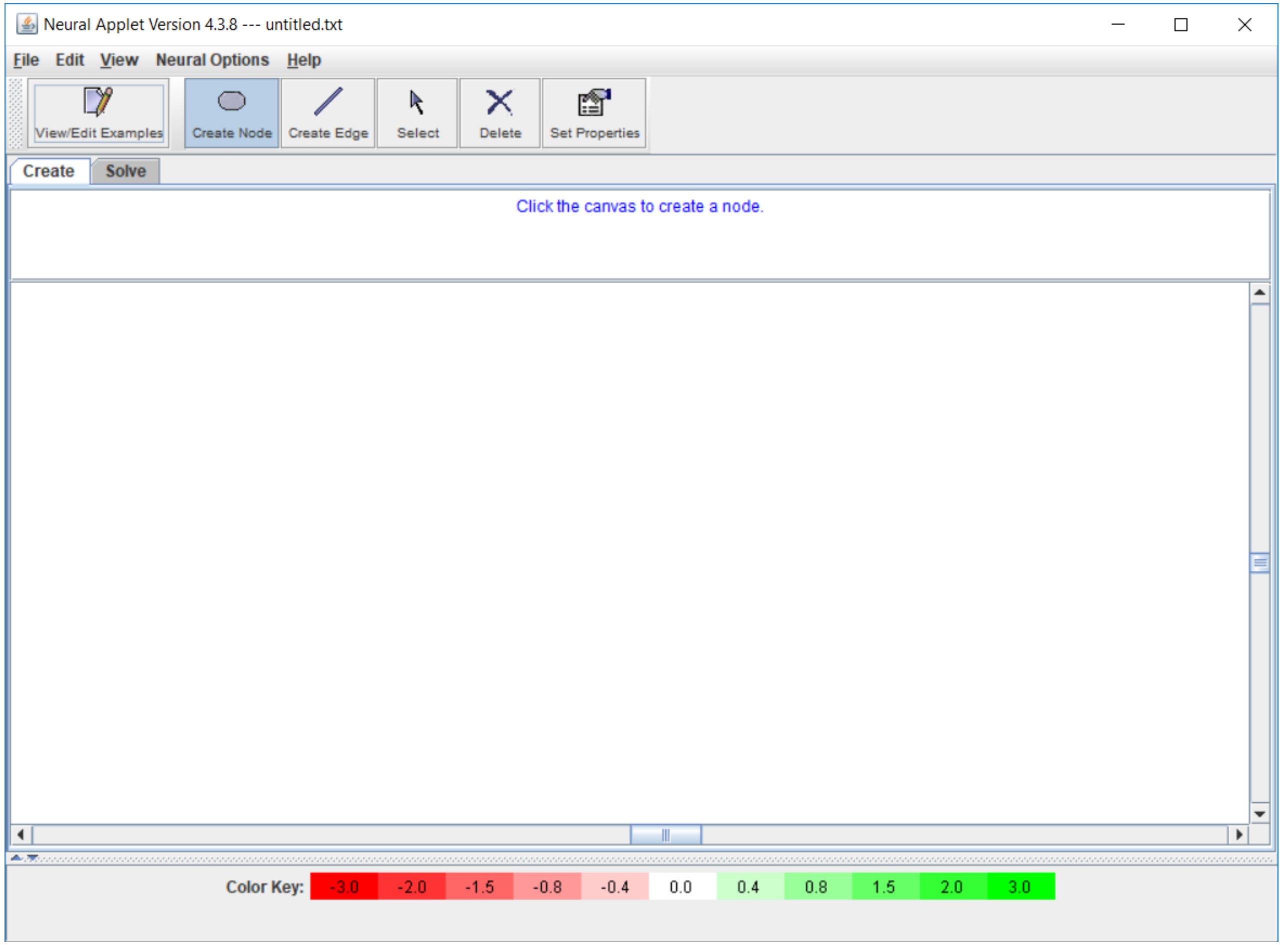
- In the folder **\JustNN\Samples** you will find many other examples.
- Moreover, in the page from which you downloaded JustNN there is a short user manual.

Useful concepts

- https://en.wikibooks.org/wiki/Artificial_Neural_Networks/Neural_Network_Basics#Momentum
- A practical example:
- https://www.facebook.com/quartznews/videos/1057379764295725/?fref=nf

A simulation tool

- AIspace (www.airspace.org)
- Create a new ANN
- select 'Create New Graph' from the 'File' menu

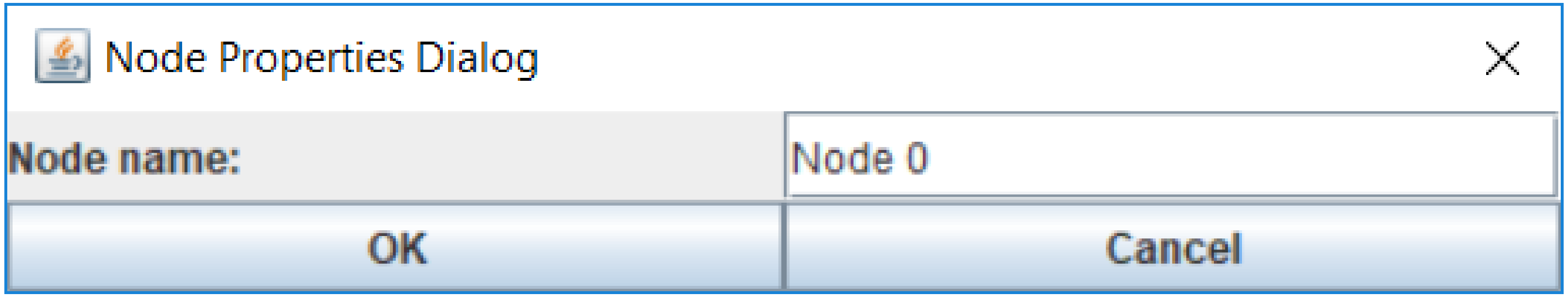


Click the canvas to create a node.

Prof. R. Folgieri

Create the nodes

- left click on the blank white canvas to create a node
- when a button is clicked, the message panel above the canvas displays a message giving you information on of what you can do next



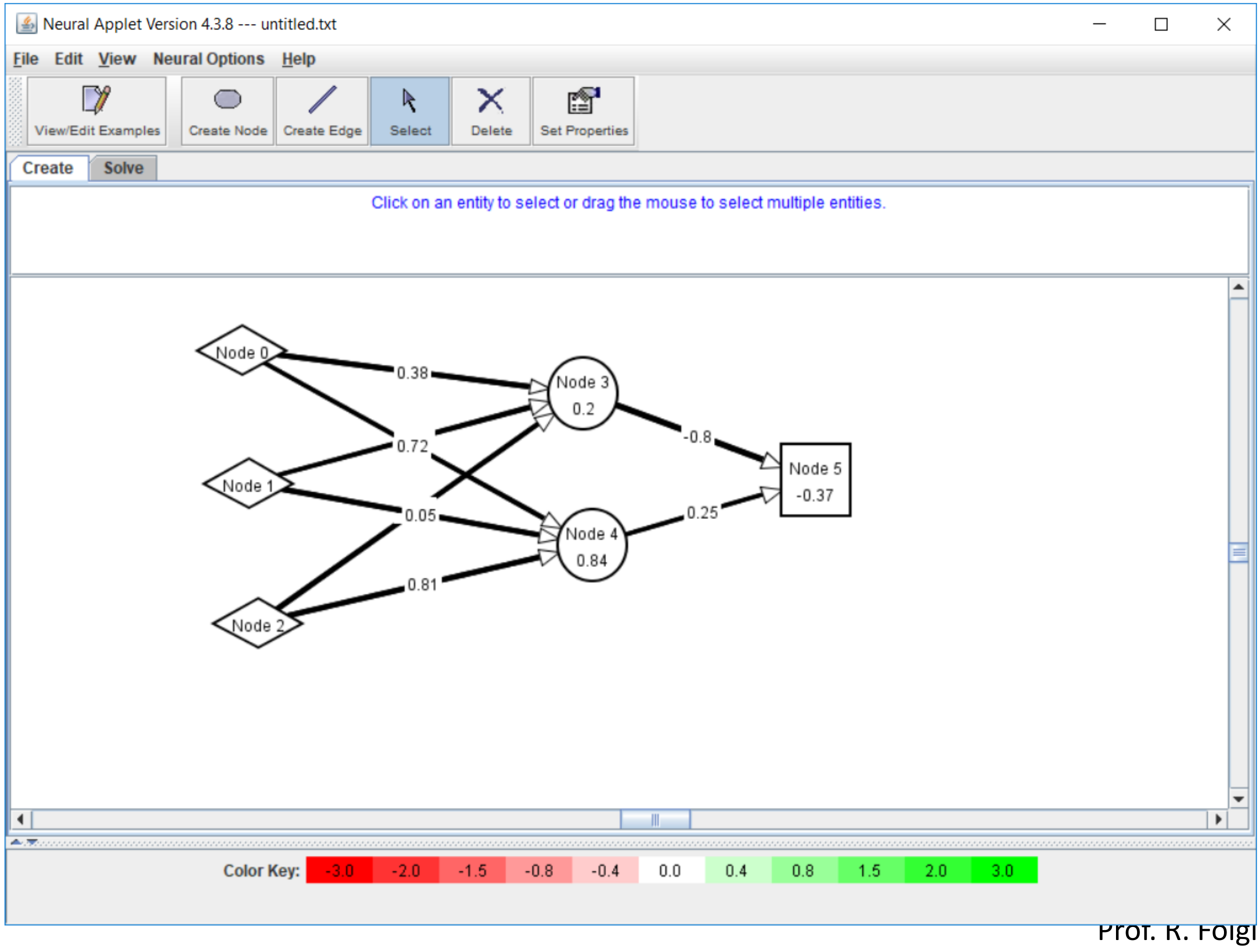
Node Properties Dialog

Node name: Node 0

- In this case, in the dialog box, you can set the node name (ie the name of the input variable)

Connect the nodes

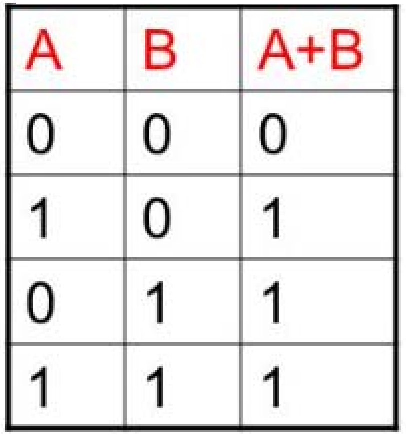
- After create the nodes, connect them
- the applet automatically detects whether a node is an input, output, or hidden node once you add edges and connect your nodes together


Click on an entity to select or drag the mouse to select multiple entities.

Insert values

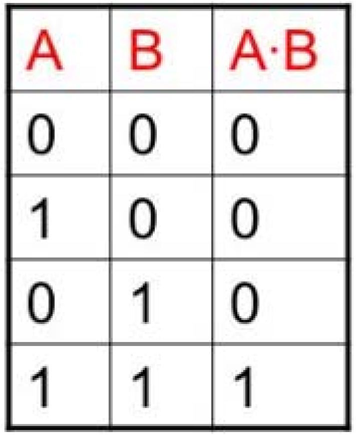
- Now we need the training set and the test set (control).
- Add examples to your training and test sets, by clicking View/Edit Examples.
- To add a new example, click on the Add New Example button. Add values.
- Imagine you want to evaluate the result of: **(X and Y) or Z** where x is node 0, y is node 1, z is node 3

OR



A	B	A+B
0	0	0
1	0	1
0	1	1
1	1	1

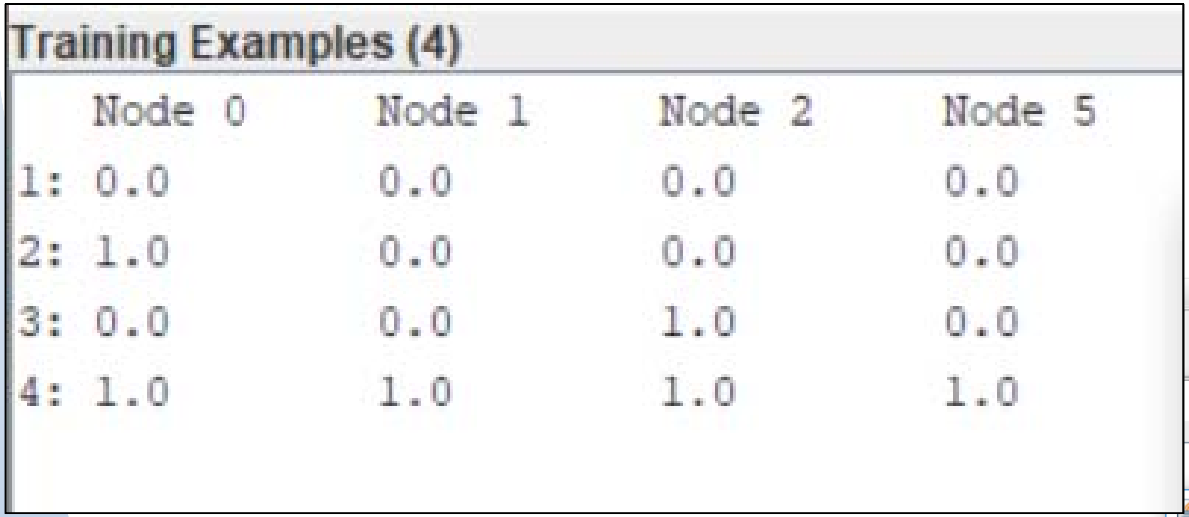
AND



A	B	A·B
0	0	0
1	0	0
0	1	0
1	1	1

Insert values

- Try with this values



Training Examples (4)

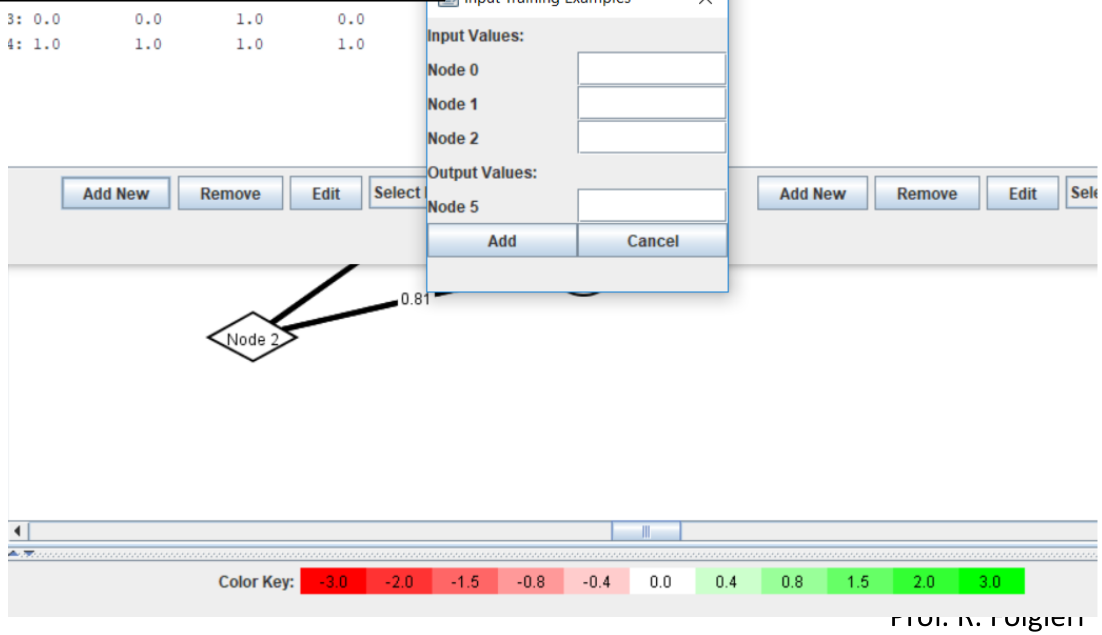
	Node 0	Node 1	Node 2	Node 5
1:	0.0	0.0	0.0	0.0
2:	1.0	0.0	0.0	0.0
3:	0.0	0.0	1.0	0.0
4:	1.0	1.0	1.0	1.0

OR

A	B	A+B
0	0	0
1	0	1
0	1	1
1	1	1

AND

A	B	A·B
0	0	0
1	0	0
0	1	0
1	1	1



Test Examples (0)

```
NO DATA
```

```
3: 0.0      0.0      1.0      0.0
4: 1.0      1.0      1.0      1.0
```

Input Training Examples

Input Values:

Node 0

Node 1

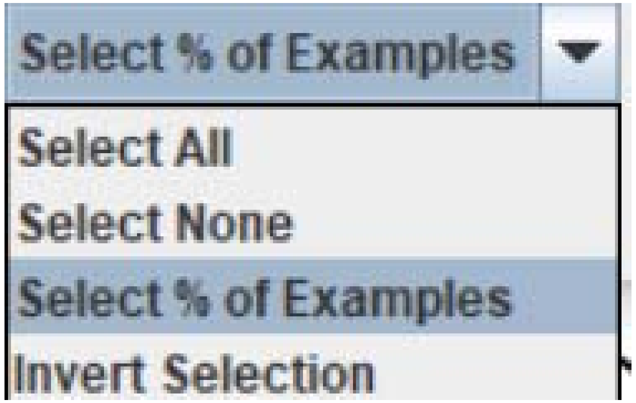
Node 2

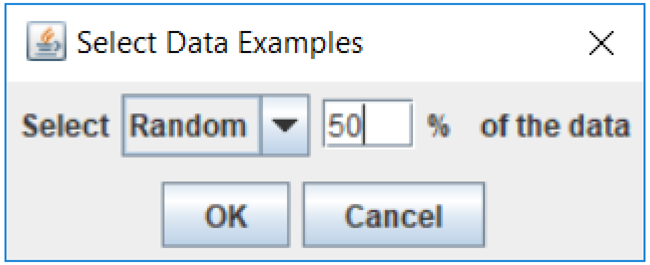
Output Values:

Node 5

Set training and test examples

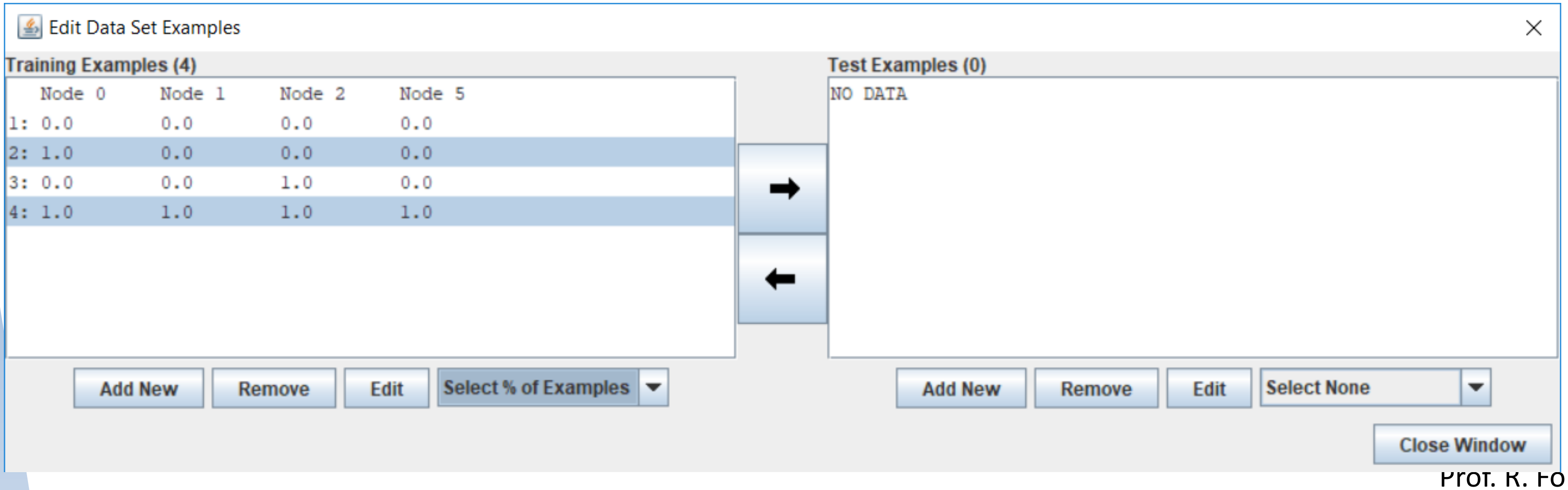
- Now select 'Select Percentage of Examples' on the training example side. Choose 50% of the training examples (Random or First or Last).





Select Random 50 % of the data

- Move the select record on the right using the arrows.



Training Examples (4)

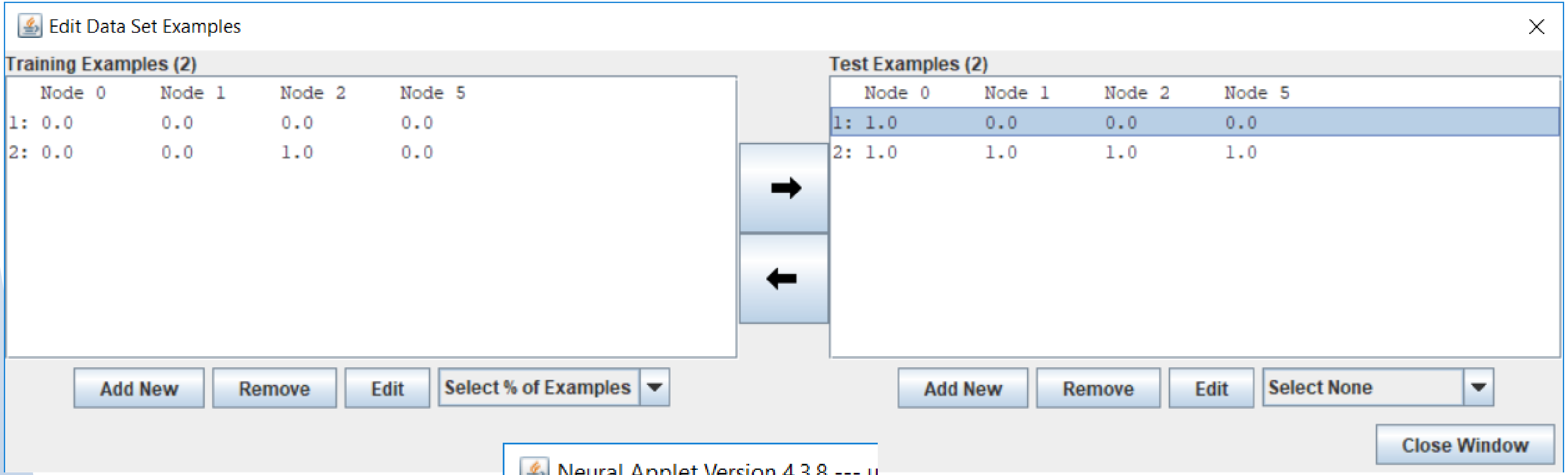
```
    Node 0     Node 1     Node 2     Node 5
1:  0.0        0.0        0.0        0.0
2:  1.0        0.0        0.0        0.0
3:  0.0        0.0        1.0        0.0
4:  1.0        1.0        1.0        1.0
```

Test Examples (0)

```
NO DATA
```

Save the ANN

- This is what you will see



Edit Data Set Examples

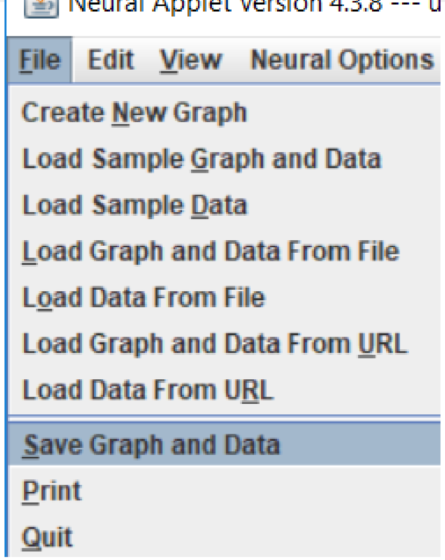
Training Examples (2)

	Node 0	Node 1	Node 2	Node 5
1:	0.0	0.0	0.0	0.0
2:	0.0	0.0	1.0	0.0

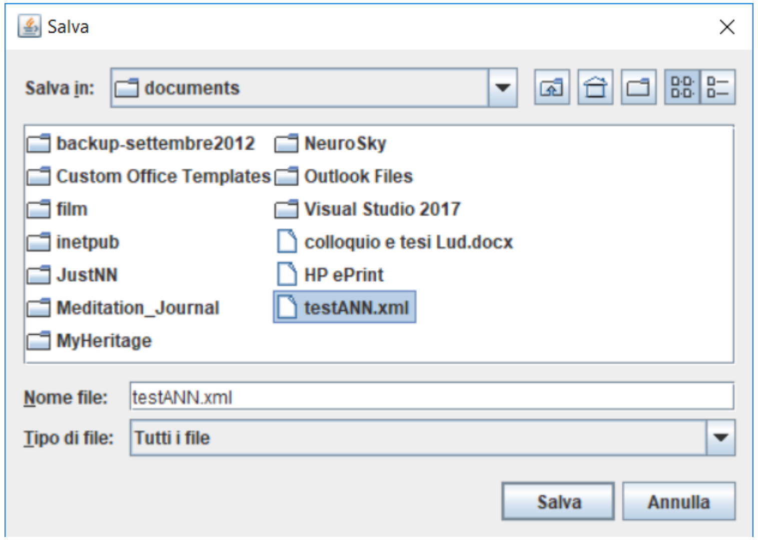
Test Examples (2)

	Node 0	Node 1	Node 2	Node 5
1:	1.0	0.0	0.0	0.0
2:	1.0	1.0	1.0	1.0

- Save your ANN



Neural Applet Version 4.3.8 --- u



Salva

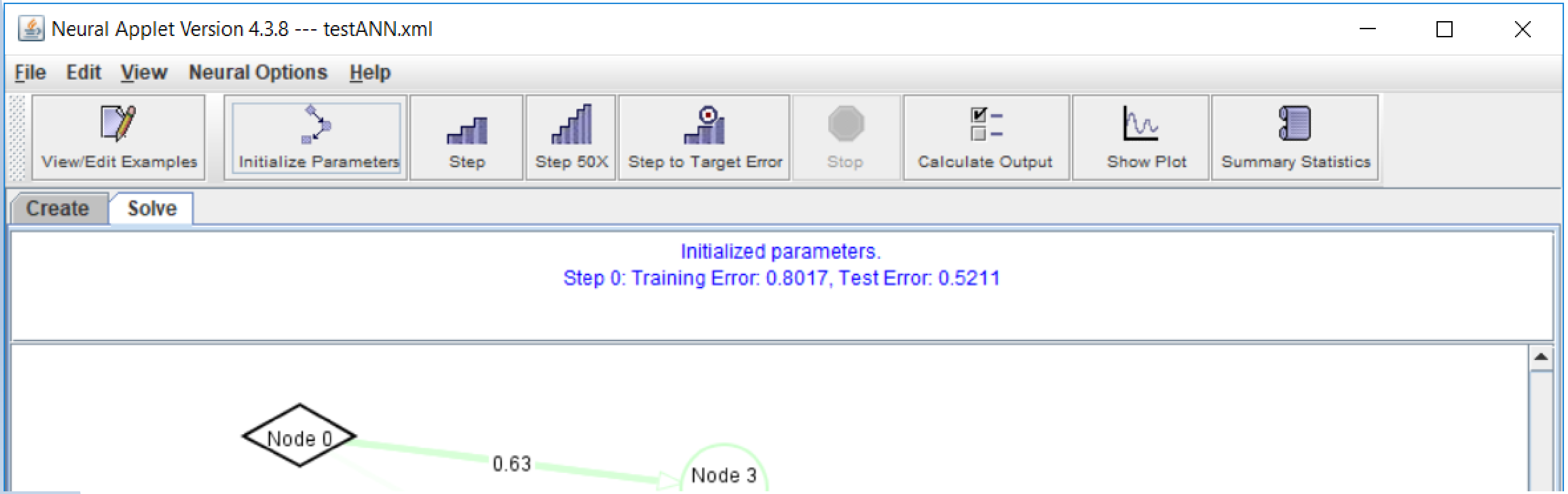
Salva in: documents

Nome file: testANN.xml

Tipo di file: Tutti i file

Run the ANN

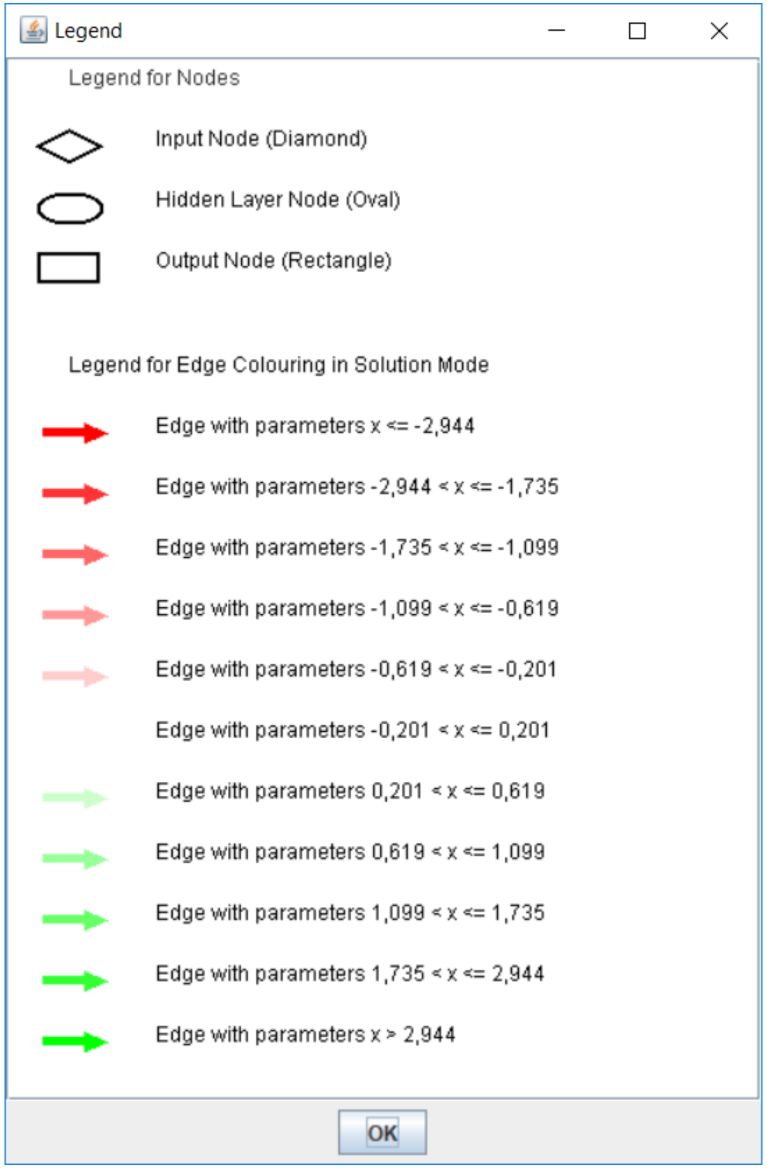
- Go in the SOLVE tab.
- Click on Initialize Parameters to reset the learning (default, random values in the range [-1,1] to parameters).
- If you want to customize them, you can click on 'Parameter Initialization Options...' (Neural Options menu).



Initialized parameters.
Step 0: Training Error: 0.8017, Test Error: 0.5211

Run the ANN

- Nodes and edges may change colors corresponding to their values.
- 'Help' menu, click on 'Legend for Nodes/Edges' to show the legend.
- To modify the ANN, click on tab 'Create' to return to create mode



Legend for Nodes

◇ Input Node (Diamond)

⬭ Hidden Layer Node (Oval)

▭ Output Node (Rectangle)

Legend for Edge Colouring in Solution Mode

→ Edge with parameters x <= -2,944

→ Edge with parameters -2,944 < x <= -1,735

→ Edge with parameters -1,735 < x <= -1,099

→ Edge with parameters -1,099 < x <= -0,619

→ Edge with parameters -0,619 < x <= -0,201

Edge with parameters -0,201 < x <= 0,201

→ Edge with parameters 0,201 < x <= 0,619

→ Edge with parameters 0,619 < x <= 1,099

→ Edge with parameters 1,099 < x <= 1,735

→ Edge with parameters 1,735 < x <= 2,944

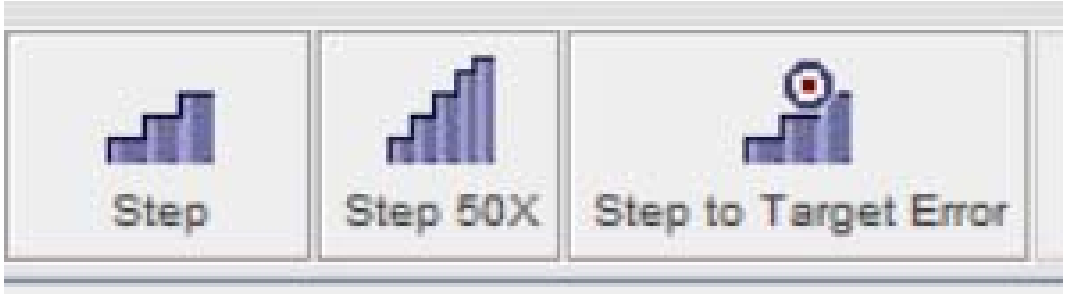
→ Edge with parameters x > 2,944

Backpropagation ANN

Three ways to perform backpropagation:

1. one step at a time → click the **Step** button.
2. 50 steps at a time → click the **Step 50X** button.
3. Run the all the ANN steps till a preset minimum error is achieved (0.1 by default) → click the **Step To Target Error** button.

Try to see the difference in our ANN...

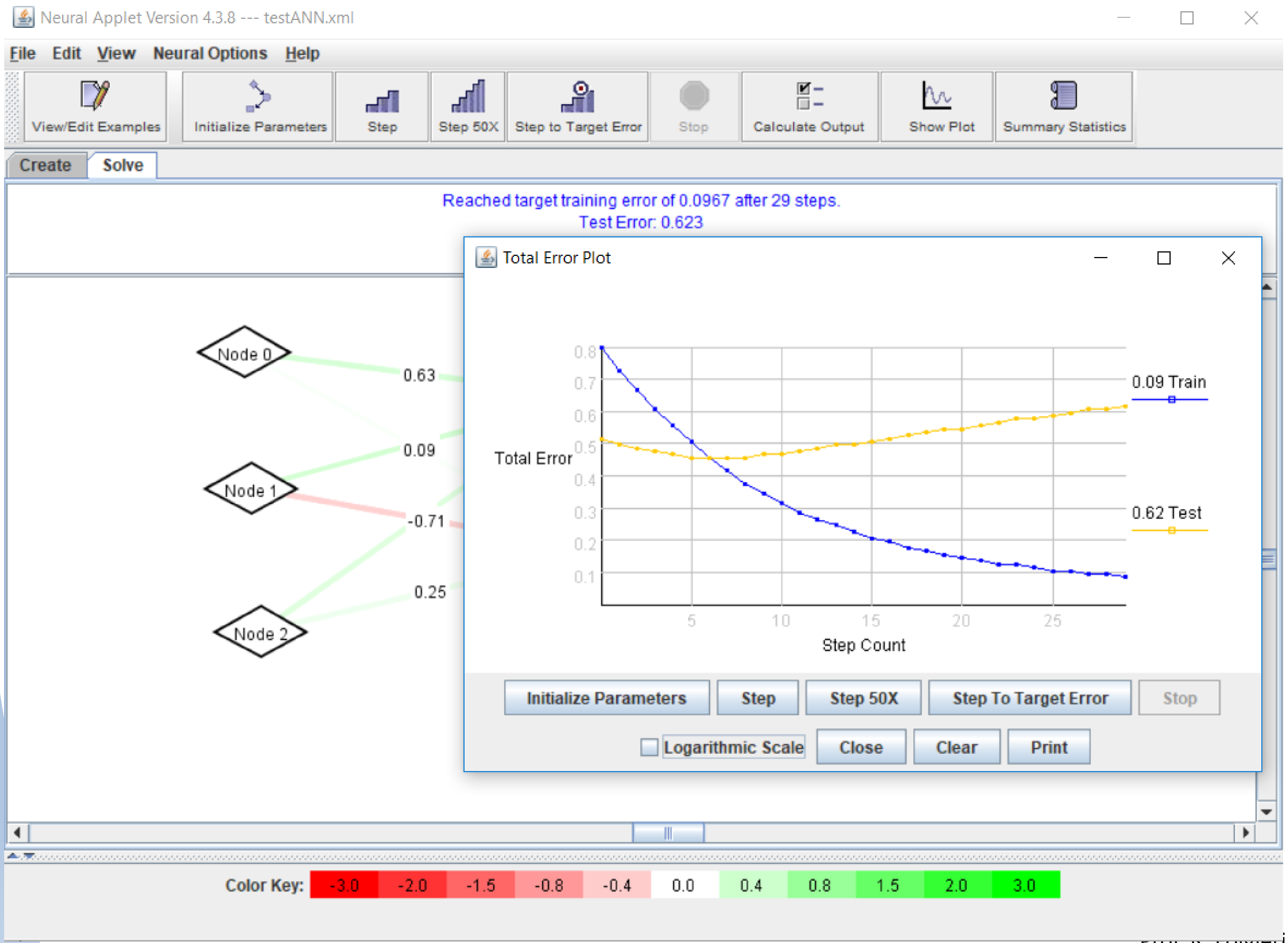


Message at the top of the network canvas: how many steps you have currently run and error for training and test examples. You can decide to stop.

Error Plot

- Click the **Show Plot** button.
- blue plot: training set error
- orange plot: test set error.
- You can see the error has been minimized.
- The plot window also shows:
  - Initialize Parameters, Step, Step X, Step to Target Error, Stop buttons.
  - are buttons to close, clear, and print the plot window and a checkbox to switch between logarithmic and standard display modes.

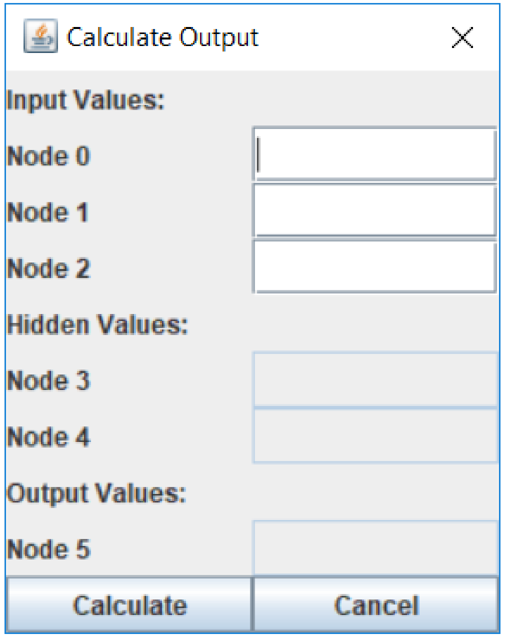
Error Plot



Reached target training error of 0.0967 after 29 steps.
Test Error: 0.623

Calculate the output

- To get an output given a set of inputs, click the **Calculate Output button** and enter the desired inputs.
- The given inputs are NOT added to the test or training sets and do not affect the learning of the network in any way.



Input Values:

Node 0	
Node 1	
Node 2	

Hidden Values:

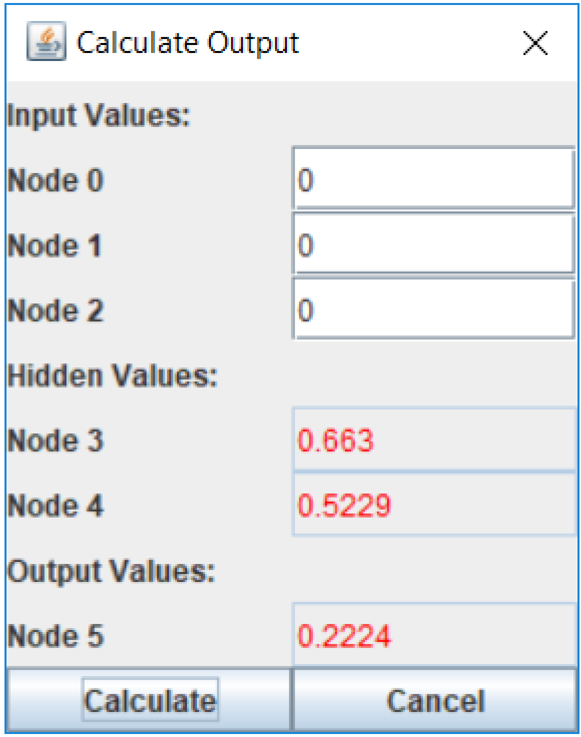
Node 3	
Node 4	

Output Values:

Node 5	

Try for some value.

For example ->



Input Values:

Node 0	0
Node 1	0
Node 2	0

Hidden Values:

Node 3	0.663
Node 4	0.5229

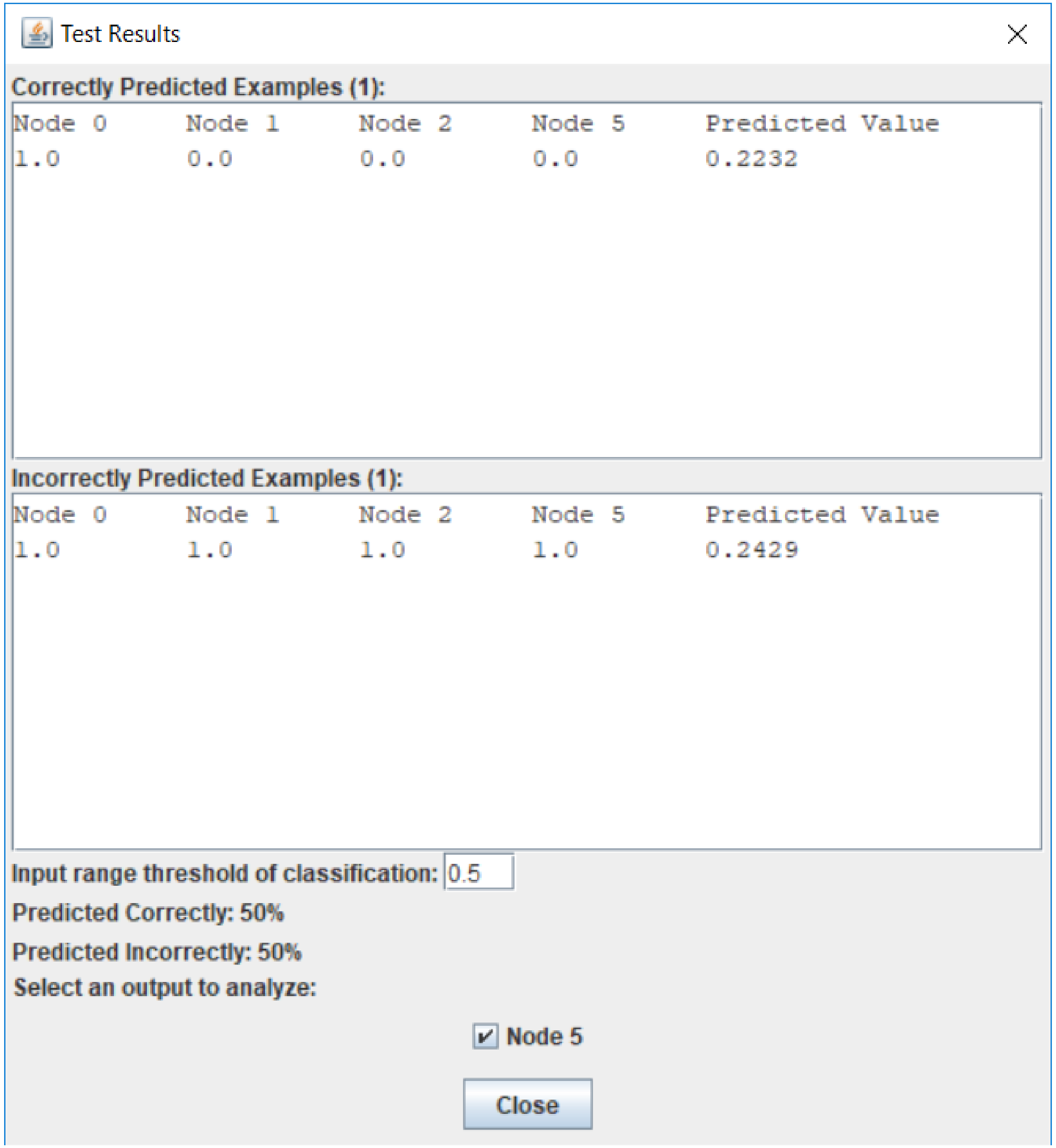
Output Values:

Node 5	0.2224

Summary statistics

- Statistics for the test set:
  - all the test examples as a table, with the predicted value,
  - classified as correct or incorrect depending on a classification range determined by the user (a threshold, by defaults = 0.5),
  - a percentage correct or incorrect.
- You can also select which output's predicted value is displayed in the table (if you have more than one output, of course).

Summary statistics



Correctly Predicted Examples (1):

```
Node 0      Node 1      Node 2      Node 5      Predicted Value
1.0         0.0         0.0         0.0         0.2232
```

Incorrectly Predicted Examples (1):

```
Node 0      Node 1      Node 2      Node 5      Predicted Value
1.0         1.0         1.0         1.0         0.2429
```

Input range threshold of classification: 0.5

Predicted Correctly: 50%

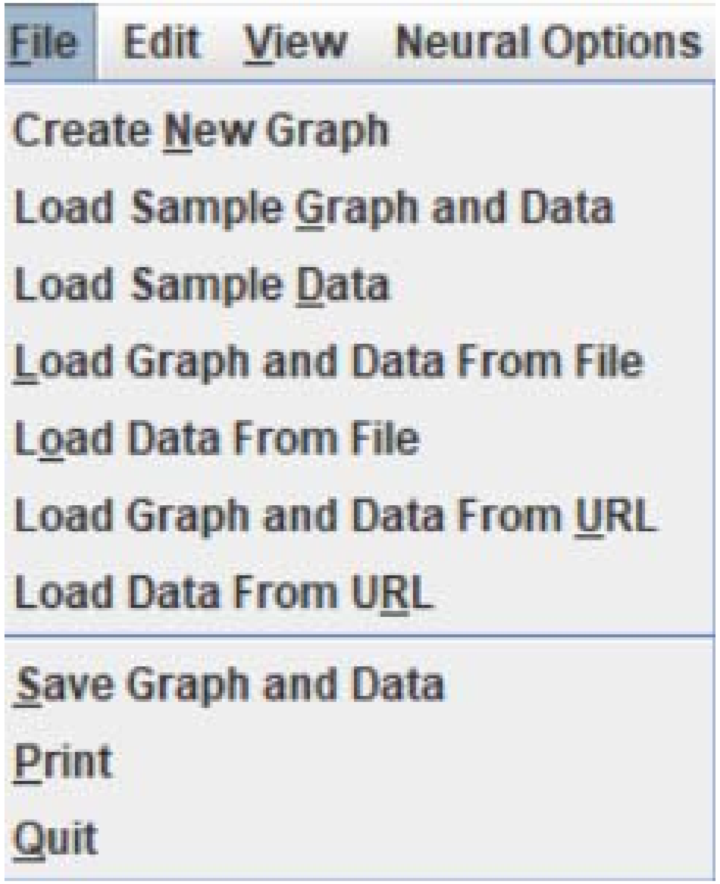
Predicted Incorrectly: 50%

Select an output to analyze:

☑ Node 5

Load data, graph and preexisting ANNs

- From menu FILE



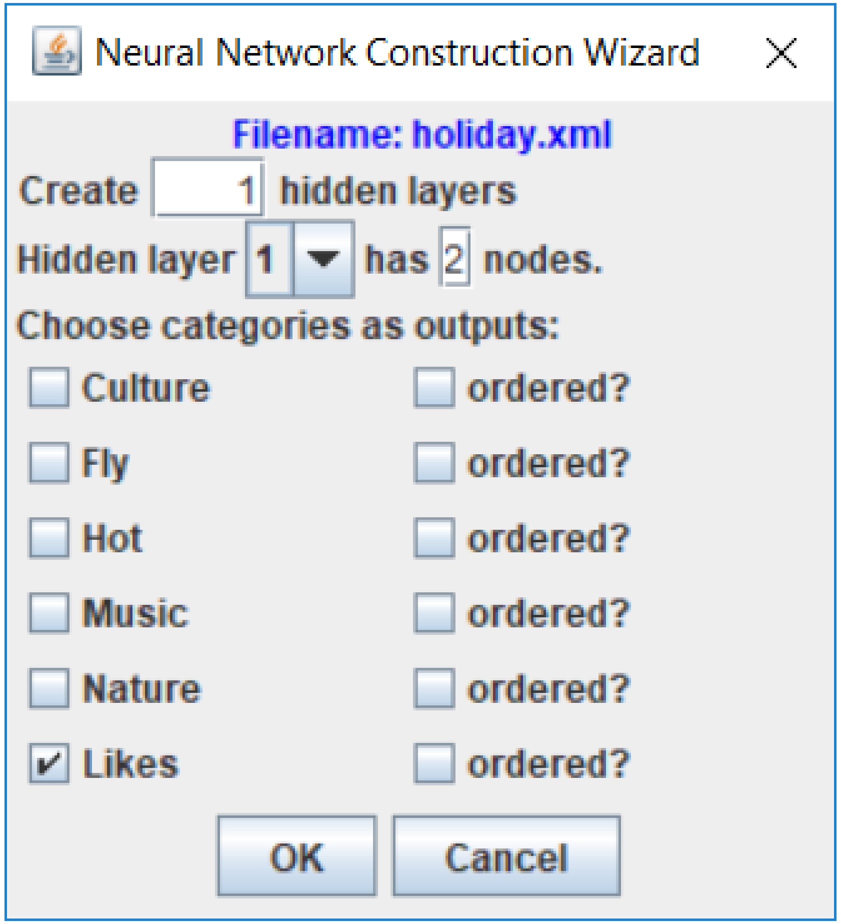
- If you want to load data from a file, select LOAD DATA FROM FILE (comma-delimited data)

Load data, graph and preexisting ANNs

- You can also load a sample data file ('File' menu and select 'Load Sample Data‘).
- The Construction Wizard dialog will query you for information on the neural network you want to build:
  - the number of hidden layers needed,
  - the number of nodes for a specific hidden layer.
- The number of nodes default to 2.

Let’s try

- Try with a sample dataset, for example with the dataset named HOLIDAYS



Filename: holiday.xml

Create 1 hidden layers

Hidden layer 1 has 2 nodes.

Choose categories as outputs:

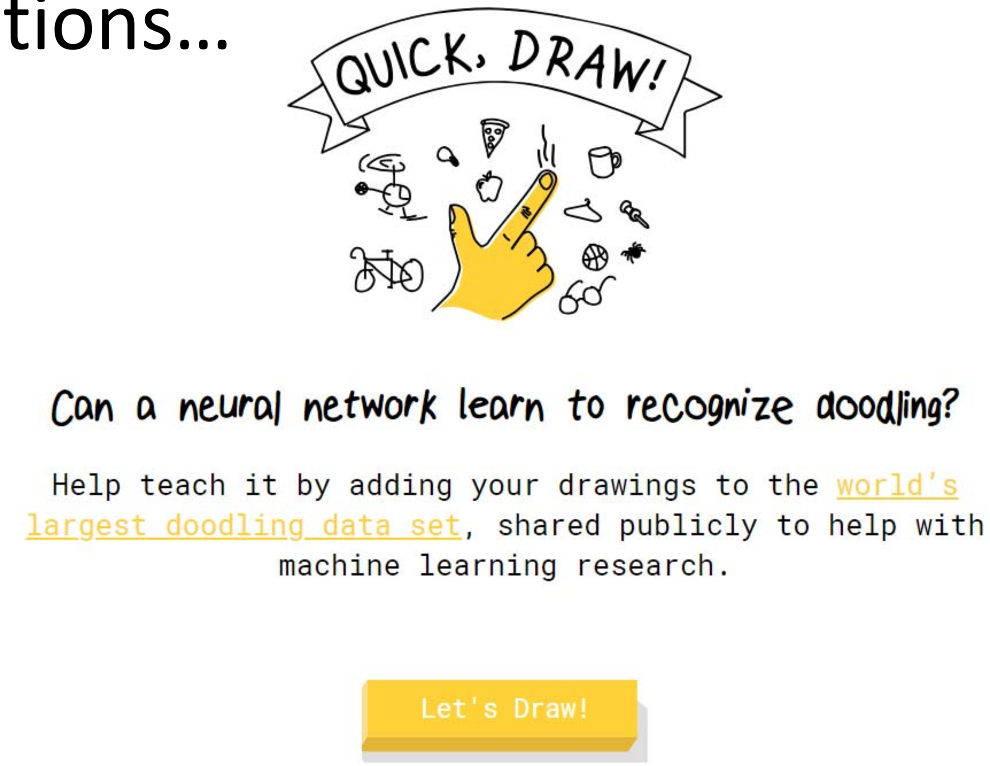
□ Culture	□ ordered?
□ Fly	□ ordered?
□ Hot	□ ordered?
□ Music	□ ordered?
□ Nature	□ ordered?
☑ Likes	□ ordered?

Dynamic outline

- Introduction
- Taxonomy
- AI fundamentals: Philosophy, History
- GOFAI vs modern AI
- Practical examples:
  - Turing Test and Chatbots
  - ANNs (machine learning)
  - BCIs ← NEXT LESSON
  - Ontologies (depending on the time)
  - Philosophy of AI & Ethic issues

Try a neural network

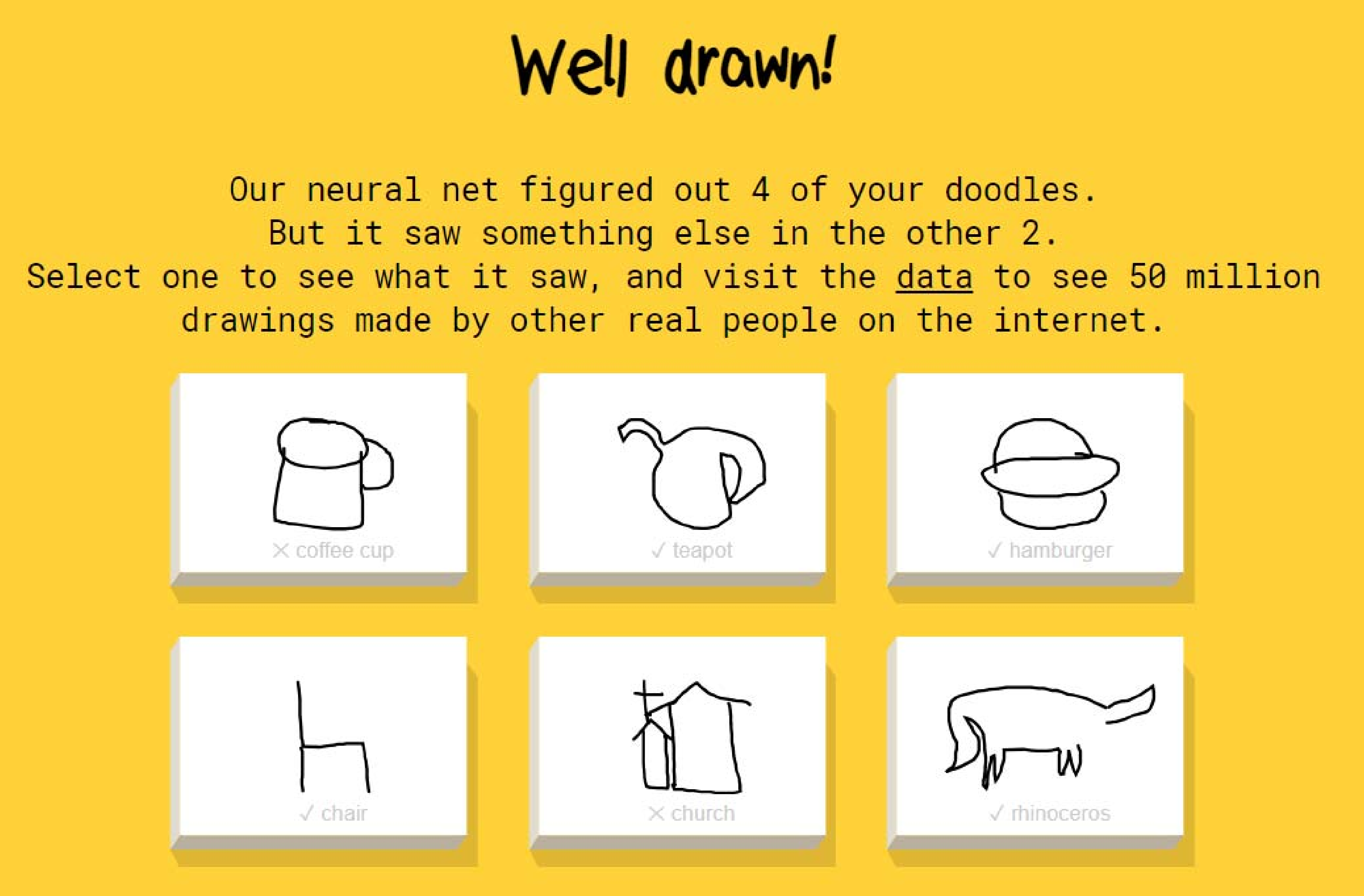
- To recognize drawing…
- Go to https://quickdraw.withgoogle.com/
- Select ENGLISH (bottom right popup menu)
- Follow the instructions…



Can a neural network learn to recognize doodling?

Help teach it by adding your drawings to the world's largest doodling data set, shared publicly to help with machine learning research.

See you next time…



Well drawn!

Our neural net figured out 4 of your doodles.
But it saw something else in the other 2.
Select one to see what it saw, and visit the data to see 50 million drawings made by other real people on the internet.

✗ coffee cup

✓ teapot

✓ hamburger

✓ chair

✗ church

✓ rhinoceros

Raffaella.Folgieri@unimi.it