INTRODUCING NEURAL STUDIO: AN ARTIFICIAL NEURAL NETWORKS SIMULATOR FOR EDUCATIONAL PURPOSES

Malek Adjouadi, Ph. D., Director
Melvin Ayala, Ph.D.
Center for Advanced Technology and Education
Electrical & Computer Engineering Department,
Florida International University
10555 W. Flagler Street, Miami, FL, 33174
Email addresses: Adjouadi@fiu.edu, Melvin.Ayala@fiu.edu

ABSTRACT

This work has been motivated by the increasing effort currently required in educational institutions while using computational tools for teaching Artificial Neural Networks (ANN). An appropriate and user-friendly programming tool is proposed with the aim to redress the situation in this important information science discipline. The programming environment is established with added capabilities over professional packages. The programming tool named *Neural Studio* is a user-friendly solution for the Windows platform developed at the Electrical & Computer Engineering department at Florida International University, which allows designing and studying the behavior of ANNs without the need for program-code writing. The interface provides graphic modules for network design, a pattern editor, a backpropagation training module as well as modules for feature maps, clustering, and memory association, among other possibilities. The software was written in Borland Delphi, which uses object PASCAL code, a high level programming language with an easy syntax and created for educational purposes.

INTRODUCTION

Artificial Intelligence (AI) with its many corollaries has become a key technology in many of today's novel applications in the design of intelligent machines, especially intelligent programs for achieving specific goals in ways that are computationally effective. New areas of information processing and understanding (fuzzy logic,[1] evolutionary computation and genetic algorithms,[2] expert systems[3] and Artificial Neural Networks (ANN)[4]) often become powerful resources for solving practical problems, such as pattern recognition, association and classification, forecast studies, and control applications, etc. Even commercial products with some kind of AI-value added, ranging from industrial machines, cars, up to household appliances have now found wide acceptance in the market place.[5]

Among all the disciplines of AI, which are all challenging, ANNs is the one discipline that is most intriguing. This is in part because ANN designs are an attempt at modelling the functionality of the human brain and its neuronal connectivity. It should also be noted that since 1958, when Rosenblatt introduced the perceptron,[6] the theory of ANN has been substantially enriched, both theoretically and in the domain of applications.

A widely accepted principle in pedagogy is that effective learning can be achieved by training.[7] With only a few exceptions, most of the examples needed to cover the most important topics of ANNs require working with considerable amounts of data so that a problem can be understood and its solution interpreted. With the computational requirements comes the need for programming skills in neural networks required to automate the analysis and

interpretation of these large data sets. The objective behind the development of *Neural Studio* is to remove the programming language requirement in the effective teaching of ANNs. To this end, lecture time allocation and laboratory training for the students are improved. Skilled programmers will find in *Neural Studio* a platform for advancing their skills in real-world applications.

In this information era, the theory of ANNs can only be efficiently applied to practical problems with the use of computers. Therefore, considerable amounts of programming tools have been developed and can be found in universities, industry, and in the marketplace as a whole. The list of packages available is extensive.[8]

Observations made on ANNs course outlines from different universities led to the conclusion that MATLAB's ANN toolbox[9] is the most prevalent tool in the teaching of this discipline.[10] Unfortunately, this toolbox requires knowledge about a scripting language that is quite similar to the C language.[11] Teaching ANNs with this tool requires students to acquire knowledge of a programming language, which adds to the level of difficulty in the learning process. Consequently, the criteria of avoiding the requirement of a programming language in teaching ANNs is not yet met with MATLAB.

After conducting research on other available tools, the following issues were noted by the authors:

- **There are no tools for unlicensed use in academia,**
- **There are not enough academically oriented tools albeit licensed that are of value to the student,**
- **There is lack of information about the features and the use of the tools (users guide being difficult to read and lacking concise descriptions of desirable features),**
- **Some tools are not designed to run under the Windows operating system (often opting for UNIX, LINUX or other platforms),**
- **Some tools are only suited for specific areas of study (e.g., economics, business),**
- **No open source tools have been found.**

An assessment of desired features for an educational tool could be summarized as follows:

- **Convenient and user-friendly interface: The windows should allow users to interact with the system primarily through mouse clicks, making keyboard inputs only occasionally.**
- **Removed requirement for programming: This feature is necessary to remove the constraints imposed by powerful tools such as MATLAB, which provides a neural networks toolbox but still requires knowledge about a pseudo-language based on C.**
- **Open source: Easy to understand source code of key calculation processes must be made available from within the application.**
- **Demonstration modules: Examples on how to use the programming tool must be given.**
- **Freeware: Enhanced accessibility for educational institutions must be provided.**
- **Compatibility with Windows operating system: The software designed needs to run under the Windows operating system, which is widely used in academia.**
- **Stopping options: Options to stop the calculations at specified points of the algorithms need to be included (to allow viewing intermediate results to test for convergence issues or for additional debugging opportunities).**
- **Stand-alone application: The stand-alone application ensures that the tool box is not dependent of another application.**
- **Configurable display: Configurable display capabilities (graphics, charts, plots, and options to turn the software**

into a high-speed calculation tool) should be made available.

Seeking to realize these aforementioned features as described above in an integrated teaching tool for ANNs was precisely the motivation for developing Neural Studio. In subsequent sections, this integrated programming tool will be described and special emphasis will be given to its enhanced features.

NETWORK DESIGN WITH NEURAL STUDIO

The programming tool was primarily developed for educational purposes, for use in postgraduate teaching. Further improvements were made, especially in speeding up calculations and providing different types of graphical outputs. In retrospect, the programming platform that was chosen for developing the application was Borland Delphi,[12] which uses object PASCAL,[13] a code easy to understand. The Delphi language was deemed appropriate for this work because it provides a large library of efficient components for creating charts and other graphical outputs. The compiled Delphi code is also fast and computationally efficient under the proposed design configuration.

Neural Studio's main window is illustrated in Figure 1 and is designed to contain an editor for a multilayer network. It also consists of information panels, editing and processing tools, a table for network input and a corresponding results table. Program users are able to freely design the network and to customize the neurons as well as their interconnections. Interconnections can be established by drag and drop operations between neurons. Neurons are represented with circles and interconnections are represented by lines drawn between pairs of neurons.

Details of the network components (neurons and their interconnections) as well as global configuration features can be viewed and edited in the network inspector (NI). Selections can be done by clicking on components in the drawing area as well as by selecting objects from a list in the NI. Once a specific neuron or weight is clicked on, the NI changes its appearance, showing the features of the selected object. The object can also be selected using the selection box located at the top of the NI. Clicking on the drawing surface somewhere outside all neurons and weights takes the user back to the network configuration panel.

Educators and experienced users should note that for better performance, the training set should be normalized. Special attention should be dedicated to the selection of the activation functions for the output neurons during supervised training since the network outputs must be capable of reaching the declared targets. Otherwise, convergence problems, such as local minima traps, or even monotonically increasing errors, can occur.

MODULES RELATED TO SUPERVISED TRAINING

Supervised training occurs when the network outputs are compared with the targets and the resulting output errors are used as reference for updating weights and biases.

Neural Studio applies the back-propagation method[14] for the training of the feed-forward networks placed in the main window. The training is performed in a training module as shown in Figure 2, and information about how to carry out this process is taken from the network design. The data used for training is taken from the pattern module.

Graphical outputs can be customized in the training module depending on the nature of the problem addressed. For example, the simple problem of approximating a multidimensional input/output relation can be visualized by plotting a pair of input/output neurons. In classification

problems, animated charts enhance the results by showing classification regions with multiple boundaries. Errors of different types can be graphically traced along the iterations.

Perhaps the most important feature of *Neural Studio* suitable for teaching purposes is its ability to approximate different one-dimensional mathematical functions. These functions can be parameterized and used for network training. During this process, animated graphical outputs can be obtained, thus allowing program users to understand how the training process works and how the network's ability to generalize is influenced by the number of iterations.

The training module also allows customizing the output charts. Additionally, as was stated earlier, the iterations in this module can be temporarily stopped at any time, thus allowing program users to look into intermediate results, such as current status of neurons and weights.

MODULES RELATED TO UNSUPERVISED TRAINING

During unsupervised learning, the network is not trained towards specified outputs. Instead, the network seeks to find patterns or regularity in the input data. The mapping implies clustering of the data. *Neural Studio* offers two separate modules for this type of training: Kohonen Feature Maps and Clustering Networks.

The Kohonen feature map[15] is a map resulting from plotting the two-dimensional weights of the neurons in a special type of network. No supervision is performed during training. The training set can be taken from the pattern module as well as from a training set generator available in the module. Figure 3 shows a snapshot of this module.

Cluster analysis is a technique driven by the large size of the data sets involved in the solution of practical problems. This approach clusters the data together into groups, attempting to put similar ones together. This can reveal patterns hidden in the data. Predictions can then be made by comparing recent data with the different identified clusters. This type of training requires only the input and determines by itself which classes exist in the input and which input belongs to which class.

The technique used here is to calculate a multidimensional feature map where each neuron represents one cluster. As with the Kohonen feature maps, the winner-takes-all method is used. The algorithm outputs the centroids of the clusters it determines. In Figure 4, a dataset example is grouped into 7 different clusters. In a second program stage, the algorithm is recalled and all the patterns are presented to the clusters and assigned to the best matching cluster using the Euclidean distance as grouping criterion.

OTHER MODULES

Neural Studio introduces a module dedicated to memory association, which allows auto-association as well as hetero-association of patterns. This module given in Figure 5 imposes no limit on network size and lets the users choose from a list the weight adjustment rule as well as the type of activation functions. An additional module for simulating fixed weight networks such as Boltzman and Cauchy machines as well as a Support Vector Machine module is also included in the Neural Studio design.

An important condition for enhancing the teaching process is to allow program users to know the details behind the calculations, especially the utilized formulas and the step sequences related to the calculation algorithms. A *code-insight* module has been created specifically to allow users to look inside the code related to the most important program tasks. This possibility is useful for advanced program users as well as for users who want to know details about a specific

algorithm to enhance their programming skills.

Another attractive feature of Neural Studio is a small assistant placed as an on-top-window which tells new program users how to use the most important modules in an optimal fashion.

Neural Studio also offers several modules for configuration and results interpretation to constitute a comprehensive ANN teaching tool.

CONCLUSIONS

The programming tool described in this paper provides an easy-to-use tool for teaching Artificial Neural Networks. At the same time, it can be used as a powerful tool for practical studies. Many of its features make the tool suitable for teaching without the necessity for any special programming skills. Furthermore, it provides demos and customizing options which display detailed results during and after the calculation processes. At the same time, the developed program code related to the most important program tasks is available from within the application. Interested users can look into the sources and learn the details of the different algorithms used. The software package can be configured to produce high-speed calculations, a feature which makes it attractive for practical applications.

Further improvements are foreseen in the area of interface development for an open-loop control of an industrial process using a data acquisition card. A trained network via *Neural Studio* will then perform a control function. Such a tool could find usage in industrial processes with real-time requirements.

A programming package presently under development by the same authors is a neuro-fuzzy controller, which will be linked to *Neural Studio* to establish an interface for close-loop-control.

Although *Neural Studio* is far from covering all aspects of the ANN theory and application, the authors believe they have made a contribution towards the improved teaching of artificial neural networks at educational institutions.

BIBLIOGRAPHY

1. Zadeh, L.A., "Fuzzy Logics and Approximate Reasoning", Synthese 30 (1975), 407-428.

2. Vose, M.D., "Simple Genetic Algorithm: Foundations & Theory", MIT Press, 1999.

3. Agogino, A., "Introduction to Expert Systems", University of California Berkeley, 1999.

4. Haykin, S., "Neural Networks: A Comprehensive Foundation", Macmillan Co., New York, 1994.

5. Hirota, K.; Sugeno, M., "Industrial Applications of Fuzzy Technology in the World", World Scientific Publishing Company, Inc., 1995.

6. Rosenblatt, F., "The Perceptron: a Probabilistic Model for Information Storage and Retrieval in the Brain", Psych. Rev, (65), pp. 386-408, 1958.

7. Brown, H.D, "Teaching by Principles: An Interactive Approach to Language Pedagogy", Englewood Cliffs, New Jersey, Prentice Hall Regents, 1994.

8. Artificial Neural Networks: Available Software. Pacific Northwest National Laboratory [Online]: http://www.emsl.pnl.gov:2080/proj/neuron/neural/systems/shareware.html

9. "Neural Network Toolbox for Use with MATLAB", User's Guide, Version 3.0, TheMathWorks, Inc., 1998.

10. L. Shuntian, “The Analysis and Design of Systems Using MATLAB: Neural Network”, Xidian University Press, 1998.

11. Kernighan, B.W.; Ritchie, D.M., “The C Programming Language”, Prentice Hall, Inc., 1988.

12. Pacheco, X.; Teixeira, S.; Intersimone, D., “Delphi 6 Developer's Guide”, SAMS, 2001.

13. “Learn Object Pascal With Delphi”, Wordware Publishing, 2000.

14. Johansson, E.M.; Dowla, F.U.; Goodman, D.M., “Backpropagation Learning for Multi-Layer Feed-Forward Neural Networks Using the Conjugate Gradient Method”, IEEE Transactions on Neural Networks, 1991.

15. Kohonen, T., “The Self-Organizing Map", Proceedings of the IEEE, No. 78(9), pp. 1464-1480, 1990.

ACKNOWLEDGEMENTS

This research was supported by the National Science Foundation Grants EIA-9906600 and HRD-0317692, and the Office of Naval Research Grant N00014-99-1-0952.

BIOGRAPHICAL INFORMATION

Malek Adjouadi obtained his B.S. in Electrical Engineering from Oklahoma State University in 1978, his M.E. and Ph.D. degrees both from the Univ. of Florida in 1981 and 1985, respectively. Dr. Adjouadi is currently serving as Associate Professor and is founder and Director of the Center for Advanced Technology and Education established by NSF and ONR grants at the Electrical & Computer Engineering Department from Florida International University. His interests include computer vision, image processing, human computer interfaces, and applications of Neuroscience.

Melvin Ayala obtained his Bachelor in Industrial Engineering in 1984 and his Ph.D. in 1987 at the Zittau Engineering Institute, Germany. He has served as a Professor and researcher in Cuba and Brazil. Dr. Ayala is currently working as a Research Associate with the Electrical & Computer Engineering Department at Florida International University, and is currently serving as manager of the Center for Advanced Technology and Education. His fields of interest are in software development, pattern recognition, image/signal processing, artificial neural networks and fuzzy logic.

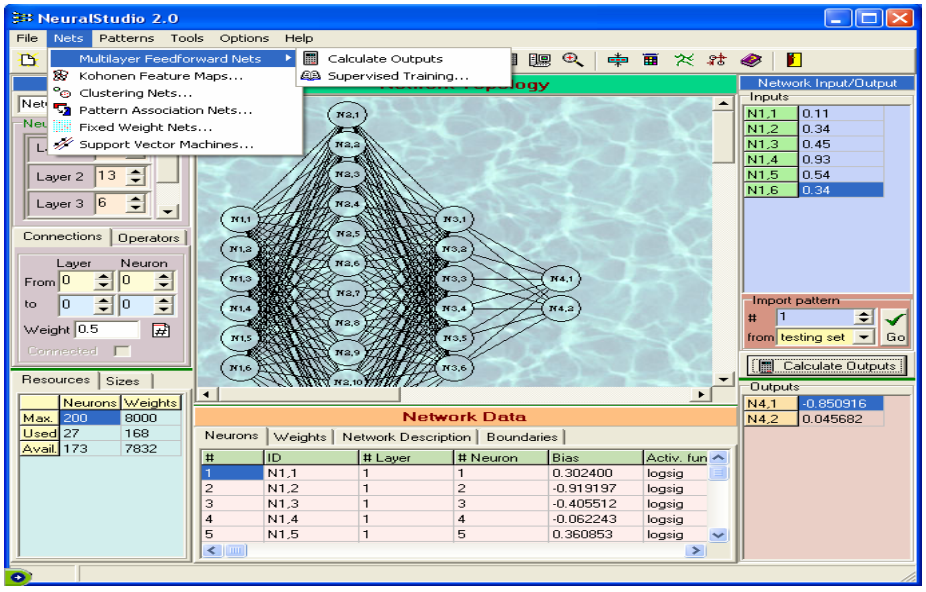

Figure 1: ***Neural Studio*****‘s main window, displaying the network editor and the input/output panel**

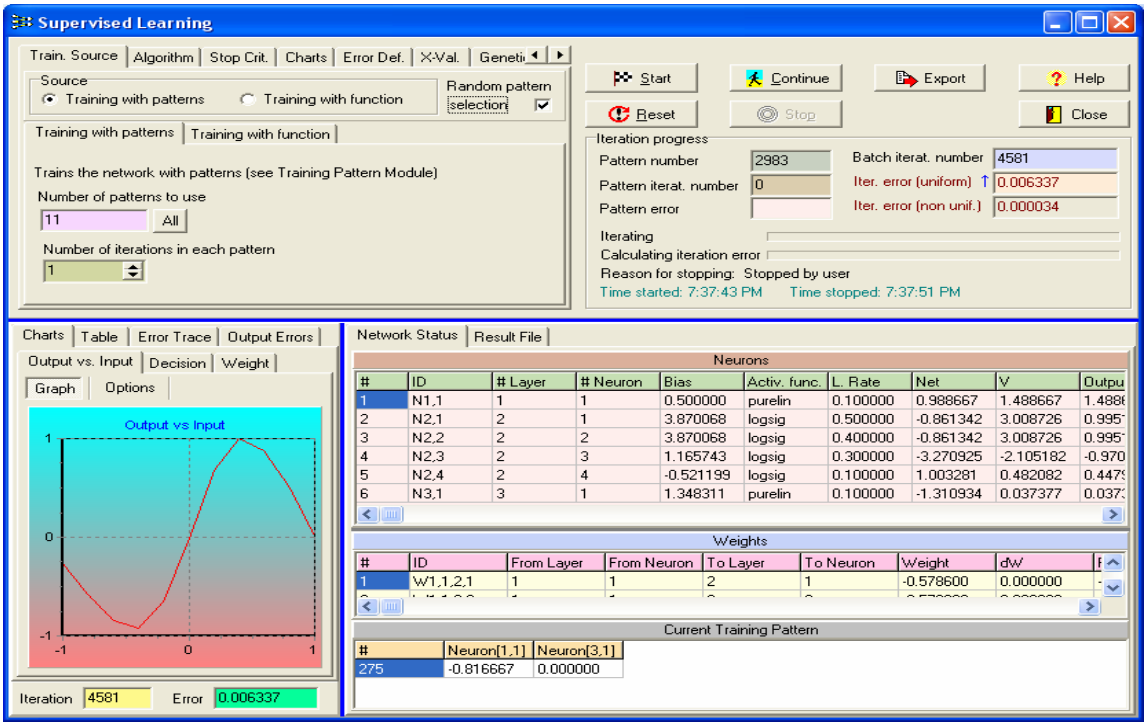

Figure 2: Snapshot of the supervised training module during a training session to approximate a parameterised sine function

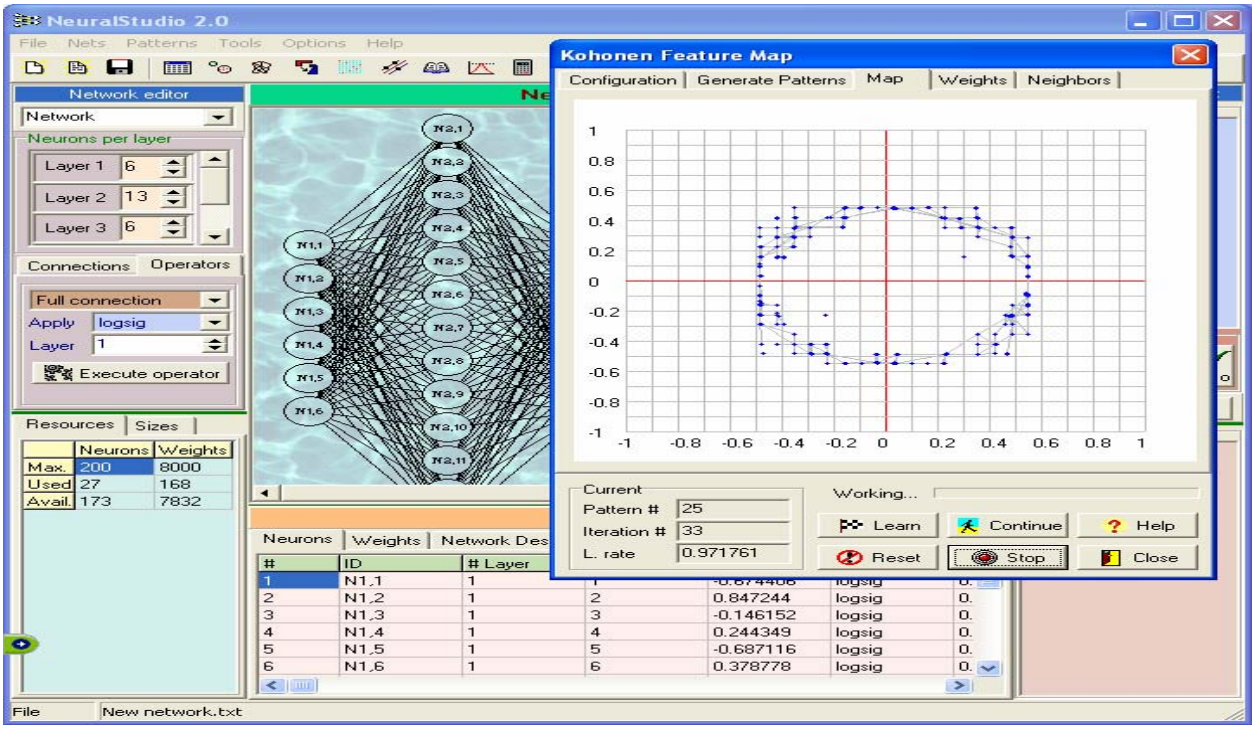

**Figure 3: Module for Kohonen feature maps analysis:
A dataset with a circular 2D distribution is being applied to the map**

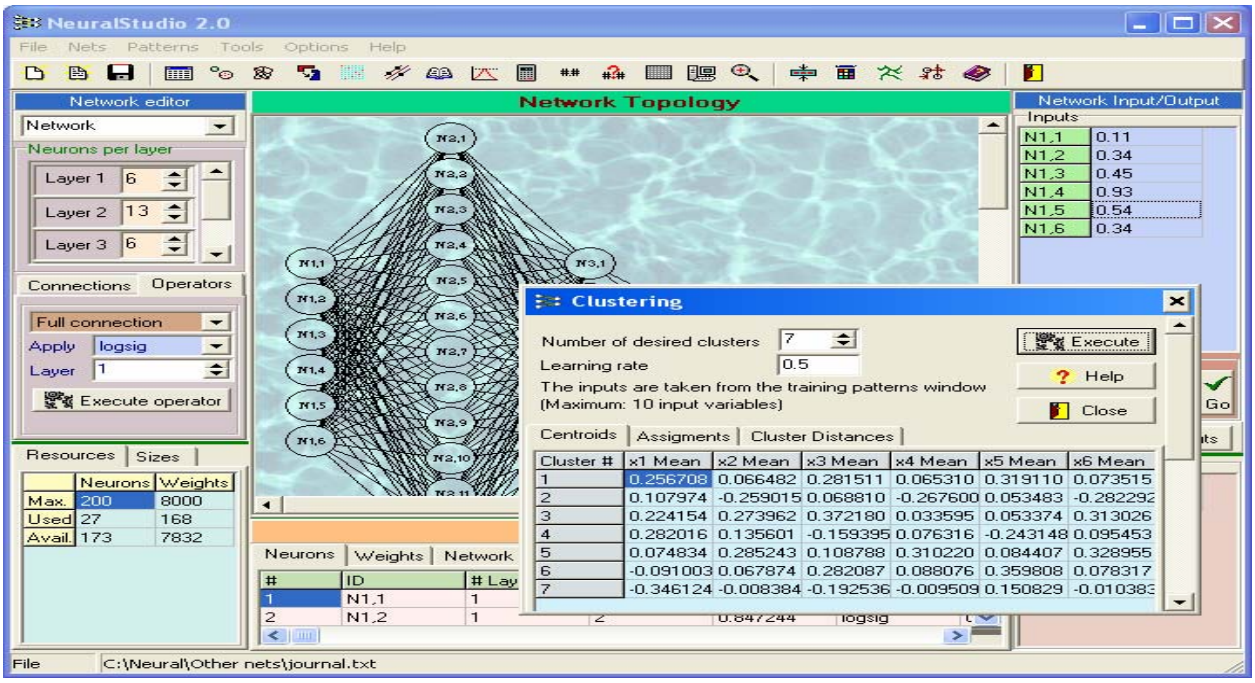

Figure 4: Neural Studio's clustering module. The training patterns are taken from the patterns module

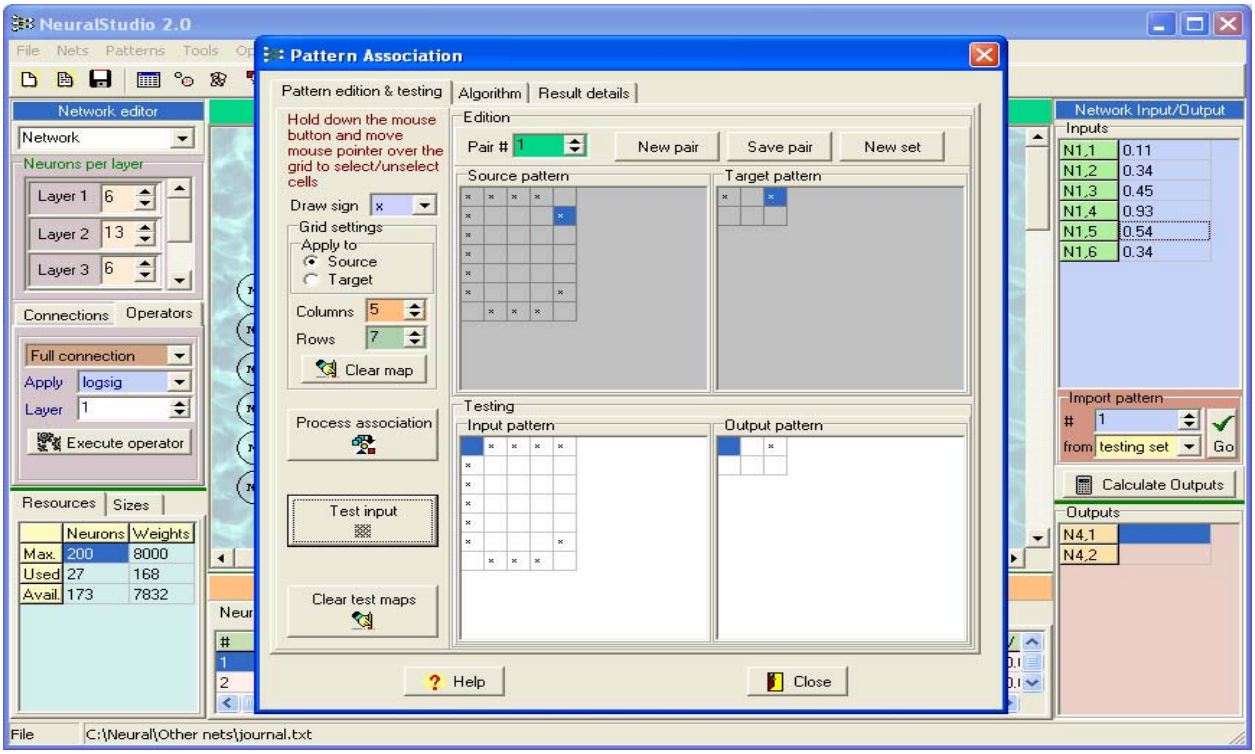

Figure 5: Module for pattern association while performing character codifications