Software Toolkit for Designing an Artificial Neural Network

M. A. Ahmad¹, J Mohamad-Saleh²

¹Electrical & Electronics Engineering School, Engineering Campus, Universiti Sains Malaysia, Seri Ampangan, 14300 Nibong Tebal, Pulau Pinang, Malaysia.
²Email: done2kl@hotmail.com, Tel.: 012-534 5055
²Email: jms@eng.usm.my, Tel.: 04-593 7788 ext 6027

1. INTRODUCTION

Artificial Neural Network sometimes referred as neurocomputers, connectionist network or parallel distributed processors (Haykin, 1994). It was first proposed several decades ago, but the interest in its use is relatively new (Lippmann, 1987). It is different from all other kinds of existing systems because of its ability to "learn" or gain "knowledge" based on a set of examples given to it. The basic idea behind neural network is to allow machine to copy the way the human brain learns, that is through experience or examples. Neural networks are composed of computational elements operating in parallel and the arrangements are reminiscent of the biological neuron of the brain. The computational elements or neurons are connected via links, whose weights that are adapted during the learning phase of a neural network model, another similarity to the biological neurons. Hence, the name of Artificial Neural Network.

ANN are generally used when most step-by-step programming fail to solve a complicated problem. Thus far, ANN has been used in many fields ranging from engineering to environmental to medicine. For example, ANN has been used to classify objects (Pope, 1994), make predictions (Simon et al., 2000; Richard et al., 2003; Billings et al., 1991; Gomm et al., 1997), recognize pattern (Karsten, 2001; Büker and Hartmann, 1996) and estimate parameter values (Mohamad-Salleh and Hoyle, 2003).

It is normal that a simulation or modeling of an ANN is first carried out to investigate its feasibility at solving a certain problem. Once the simulation is successful, then only a dedicated ANN microprocessor system is designed. One of an neural network simulation tool is called nntool (Demuth, 2002).

Although the nntool Toolbox can be used to design, train, validate and test a neural network in Matlab, the interface has not been found to be user-friendly. In nntool, for example, there are help sections for users to get information on how to use the interface but the help
does not contribute much to user. This is because all these enlightenment are given too simple explanation, that it becomes less informative for new users to understand its usage.

Basically, the processes of an ANN using nntool are not efficient for some reasons. First, the users who are not familiar with Matlab they will face a lot of difficulties in designing, testing, validate and training processes due to little proficiency in using the Matlab programming language. Most probably they will find difficulties in keeping the data, choosing the desired network models and other processes, which is involved when designing neural network because the usage of nntool interface is quite intricate. This may force a user to use trial and error methods during these processes, causing wasted time.

Second, in nntool users can look at the processes or activities that have been executed. Therefore, they have to repeat the same processes all over again for the next ANN design. If the processes are too long or too complicated, the user may forget what have been done and this could lead to failure, and waste of the user's time. Moreover, in nntool, all network commands or functions are placed in a small, convoluted and ugly interface.

Third, if the user wants to start with a new designing process using nntool, he has to delete all previous data or close Matlab and reopen it. At this stage, it shows that nntool is not sophisticated. Moreover, having to do this actually wastes a user's time.

Fourth, users are not allowed to set percentage values for testing, training and validation data sizes while designing an ANN using nntool. Hence, they have to do it by giving a command in the Matlab command window. For those who are new with Matlab, they may face difficulties in doing this.

Discussed above are the weaknesses and disadvantages of using nntool. For those users who are not familiar in using Matlab they will meet with all these difficulties. These drawbacks motivate the development of this project, whose objective is to design a toolkit, which is far more user-friendly and easy to use by any user, being at any level, compared to nntool. This new toolkit is called AnnSol.

2. METHODOLOGY AND APPROACH

This paper focuses on supervised networks i.e. Perceptron, Multi-layer Perceptron (MLP), Learning Vector Quantization (LVQ) and Hopfield. Tools such as Matlab, GUIDE and Neural Network Toolboxes are used in carrying out this project. The reason for using Matlab as the main connector to the new interface is its common use in many simulations or modeling.

There are two approaches to building this interface. One way is by using programming methods and the other is by using GUIDE. GUIDE has been used to create this interface because plenty of time can be saved using this method. Besides using GUIDE in Matlab, Visual Basic or C++ Builder can also be used to build the interface. However, it will make the task difficult particularly in creating functions for designing ANN models. Matlab already has built-in functions to design ANN, within the Neural Network Toolbox. Therefore it is a lot easier to use GUIDE to develop this interface as the necessary function can only be called from the interface programs instead of having to create new function.

GUIDE, the Matlab graphical user interface development environment, provides a set of tools for creating GUIs. These tools greatly simplify the process of laying out and programming a GUI. When GUI in GUIDE is opened, it will display the Layout Editor, which is the control panel for all the GUIDE tools. The Layout Editor enables users to layout a GUI quickly and easily by dragging components, such as push button, pop-menus, static text, checkbox, fram or axes from the component palette into the layout area. GUI tools used for this paper are:

1. Axes.
   Used to display figures, images or flatten the output result for this ANN in graphical methods such as graph
2. Frame.
   This component is used to decorate the interface windows
3. Edit Text.
   This function is used to allow users to enter data or values for creating an ANN. Command is use in this callback section, in doing so, an error message dialog box
may pop up notify users about wrong values which have been entered. For example, if an edit text requires a numeric value but a user enters an alphabet, an error message will be displayed.

A checkbox allows users to choose preferred parameters. For this paper the choices are: epoch, show, goal, hard limiter and iteration stop, which maybe needed in a training process.

5. Static text.
This function is used to display filenames when a user would like to load a file such as a data file. This function also displays the number of data in a loaded file, the processing time for training, testing and validation processes. In this paper, static text function is also used to display MSE values for testing and validation process as when all processes involved in an ANN development are completed.

6. Pushbutton.
A pushbutton is clickable. For the designed interface, a pushbutton is clicked when users would like to begin a training or adaptation process, saving, open, reset, loading, editing data or to get help on using the interface.

In nn-tool, if a user requires variable to be displayed in the Matlab workspace browser, he needs to export the variables in the export section. In contrary, AnnSol will automatically display the necessary variables once the users start a training process by clicking on the [Train] button. The “Evalin” and “assignin” Matlab functions are used to achieve this. The “Assignin” function is used to export any variable values from the GUI to the workspace while “evalin” is used to import any variable values from the workspace back to the GUI.

3. RESULTS
Below are 6 figures, which show the results of this project. Figure 1 shows the main window of AnnSol. Users can click on the pushbutton to select a desired network i.e. Multi-layer Perceptron (MLP), Learning Vector Quantization (LVQ), Perceptron and Hopfield. Figure 2 shows the MLP interface window. Figure 3 shows the LVQ interface window and Figure 4 shows the Perceptron interface window. A user is able to enter data or values, or select preferred parameters from each of the network’s interface window.

![Figure 1: Artificial Neural Network Simulator interface](image-url)
Figure 5 shows the two-dimensional Hopfield network interface and finally, Figure 6 shows the three-dimensional Hopfield network interface.
4. DISCUSSION

The *AnnSol* interface is far more user-friendly compared to *nntool*. As has been discussed in section 2, there are many weaknesses in *nntool* and all the weaknesses have been overcome by *AnnSol*. The advantages offered by *AnnSol* are:

1. With *AnnSol*, the MSE percentage for testing and validation will be shown automatically after the training, testing and validation processes are completed. This automatically not supported in *nntool*.

2. The Parameters to create a backpropagation network are very limited in *nntool*. In *AnnSol* however, all the necessary parameters are provided and can be changed as necessary by users.
3. *nntool* only provides two kinds of stopping conditions for a training process i.e. error goal and epochs. In *AnnSol* however, three kinds of stopping condition are provided. There are the validation state, error goal and epoch. In default, *AnnSol* use the validation stop condition, which this is the best stopping condition in developing a robust ANN (Bishop, 1994).

5. CONCLUSION

A new neural network simulation toolkit called the Artificial Neural Network Simulator (*AnnSol*) has been successfully developed in this project. This interface can be used hand-in-hand with Matlab to design an artificial neural network. It is capable of accomplishing processes that *nntool* can i.e. testing, training and validation whilst providing some extra functions to further facilitate the tasks.

Overall, *AnnSol* provides the convenience of creating four artificial neural network models i.e. Multi-Layer Perceptron, Learning Vector Quantization, Perceptron and Hopfield. The task of designing, training and testing a neural network is made simple by using this interface.

REFERENCES


Richard L. (2003), “Prediction of traumatic wound infection with a neural network derived decision model”, Department of Emergency Medicine, Michigan State University/Kalamazoo Centre for Medical Studies, Kalamazoo, MI