BP Network Implementation Based on Computer MATLAB Neural Network Toolbox

Lei Wang1,*
1Yunnan forestry technological college, China, 650224
*Corresponding author e-mail: tim200e@163.com

Abstract. This article introduces the practical application status and development stage of neural network toolbox and BP network from the beginning, and then analyzes the principle and characteristics of computer MATLAB neural network toolbox and BP network respectively.

Keywords: MATLAB, Neural network, BP network

1. Introduction
Neural network is to simulate the structure and operation mechanism of human brain from microstructure and function. It does not need to model the described objects and can better describe nonlinear and uncertain systems [1]. As researchers deeply study BP networks, many new fast and effective algorithms have emerged. MATLAB is based on neural network and contains a large number of function functions and algorithmic functions of BP network, which provides a convenient tool for simulation research of network [2]. Many algorithms have their own characteristics. In different case, the selection of appropriate algorithms can achieve twice the result with half the effort.

2. MATLAB neural network toolbox

2.1. MATLAB neural network toolbox overview
Based on the corresponding version of the neural network tool, we can develop a variety of activation functions using MATLAB original basic code. They are typical representative functions in the field of neural networks, and later designers simply need to call the previously developed functions directly to do the output calculation directly, no longer need to be specifically defined for the functions to be used. The tool functions it provides are very rich, some of which are generic and can be used for almost all types of neural networks. Among them, simulation function, training function, learning function, initialization function, input function and transfer function are more important. And some other functions are especially aimed at a certain type of neural network, such as the creation function of perceptron, the training function of BP network and so on. For example, for BP network model, the
toolbox provides many functions BP network analysis and design, such as creating functions and transfer functions, learning functions, display functions and so on. The functions, call formats, usage methods and precautions of the above functions are not discussed here.

2.2. Neural network training
According to the training sample database of engineering examples brought by the system, under the main interface, click the "network running" key, enter the network running interface, and then click the "network training" key, the network training begins. When the network completes the training, input a vector to the network that is not in the training set, and the network will give the output result in a generalized way. Because of the large number of groups of input vectors, it often makes the complex nested program with the general program design to appear cyclic nested loop, which makes the program time-consuming and difficult to call [3]. As a result, designers waste a large amount of time in programming, and no time to take into account the improvement of network performance. At this point MATLAB the toolbox fully shows its superiority, and all its operations are in matrix form, which makes the network training simple and clear and fast.

3. BP network model
Neurons are the most basic processing units of neural networks [4]. Neural network learning uses improved BP algorithms. The learning process includes direct computation and error propagation. Directly from the input layer to the hidden layer, the input information is calculated in stages and transmitted to the output layer. The state of each layer of neurons only affects the state of the next layer of neurons. If you do not get the desired output level, you will be forwarded to the wrong reverse forwarding process and the error signal will be returned by the original connection path. The final output of the network is similar to the output you want. Figure 1 shows the topology of a typical three-layer BP network.

![Figure 1. Schematic diagram of three-layer BP network topology](image)

4. BP principles of network design
BP layers, number of neurons in each layer, activation function, initial value, learning speed should be considered when designing the network. Theory proves that the pre-stressed network of at least one S font hidden layer and one linear output layer approximates any rational function which illustrates the principles of designing BP networks. That is, by increasing the number of layers, the error can be further reduced and the accuracy can be improved, but at the same time, the network can be
complicated and the training time of network weight can be increased. Another problem is that it is invalid to solve this problem without simply using a single layer network with nonlinear activation functions\(^5\). Since single-layer nonlinear networks can solve these problems well, adaptive linear networks can solve these problems well and improve the execution speed of adaptive linear networks. However, in the problem that can only be solved by nonlinear function, the accuracy of fault is not high, and only by increasing the number of segments can the expected effect be obtained. When the initial weight is too large, the weight input and \(n\) are controlled within the saturation range of the type S activation function, generating small derivatives and fitting interrupts. In order to maximize the change of the S type activation function, the weight of each neuron can be adjusted. Therefore, the random number of initial weights between \((-1, 1)\) is usually taken. In the MATLAB toolbox, this method selects only the initial value of the first hidden level and uses random numbers. The error value of the network will not deviate from the wrong interval, but will move towards the minimum error value. Therefore, usually in order to maintain the stability of the system, there is a tendency to choose a smaller learning speed. Learning rates vary from 0.01 to 0.8\(^6\). Although appropriate learning rates can be provided for each particular network, different parts of the error page may require different learning rates in more complex networks. In order to reduce the number of training and find the training time of learning rate, it is best to adopt variable adaptive learning rate, so that network training can automatically adjust the size of learning rate in stages.

5. MATLAB realization of BP neural network

5.1. Network design steps
when designing BP neural networks, the following issues should be considered: standard training parameters for training samples; (1) establishing a network: automatically determining the number of neurons at the time of shipment and at the time of shipment through sample data; the user must define the number of neurons in the hidden layer, the number of neurons in the hidden layer, the transformation function between the hidden layer and the output layer, and the algorithmic function. 2) Initialize the name to represent the initial weight on the network when creating a network tray and specify a threshold based on standard parameters. (3) Network training: Network systems represent decision-making on input parameters, target vectors, and preset sampling techniques. (4) Network simulation: test data simulation based on network capability creation.

5.2. Design examples
To verify the nonlinear mapping function of the BP network, a BP network with only one hidden layer is established in the instance to approximate the sinusoidal function. During the MATLAB instruction window, the following procedures are written:

```matlab
% Input sample data
P=0:0.1:2.*pi;
T=sin(P);

% Networking
net=newff([0,2.*pi],[5,1],{'tansig','purelin'},'trainlm');
```
% Initialization
net=init(net);

% Network training
net.trainParam.show=100; % Set training display intervals
net.trainParam.epochs=20000; % Set the maximum number of training cycles
net.trainPara.goal=0.0001; % Set performance target values
net.trainParam.lr=0.01; % Set learning factor
[net,tr]=train(net, P, T); % Network training

% Network Simulation
P1=0:0.01:2.*pi; % Select test input data
T1=sin(net, P1); % Simulation calculation
figure(1);
plot(P, '*', P1, T1) % Drawing

The fitting curve of neural network shown in Figure 2 can be obtained.

![Fit curve]

**Figure 2.** Neural network fitting curve based on sample data

Neural network fitting curve can be seen from figure 2; the trained BP network can well approximate the given objective function. Of course, users can also use the graphical user interface development tool provided by the MATLAB to design a graphical user interface based BP network application system.
6. Conclusion
MATLAB neural network toolbox has corresponding analysis and design functions of various neural network models. The designers of neural network no longer need to think about complex algorithm programming, but only need to directly call the relevant design, performance analysis, exercises, learning and other functions in the toolbox, which not only improves the designer's work efficiency, enabling them to focus on to solve other problems, but also helps to improve the efficiency of neural network design. Therefore, the MATLAB neural network toolbox has extremely high practical value.

References
[1] Luo Chenghan. BP Network Implementation Based on MATLAB Neural Network Toolbox [J]. Computer Simulation, 2004, 21(5): 109-111,115.

[2] Min Xilin, Liu Guohua. To develop BP network application MATLAB neural network toolbox [J]. Computer Applications, 2001, 21(0 z1): 163-164.

[3] Pan Hanming, Huang Jian, Wang Yaohong. The Construction of Basic Selection System Based on MATLAB Neural Network Toolbox [J]. Journal of Wuhan University of Technology, 2002, 24(12): 52-53.

[4] Frees Technology Product Development Center. Neural network theory and MATLAB7 realization [M]. Beijing: Electronic Industry Press, 2005: 44.

[5] Lan Xuemei, et al. Matlab implementation of BP network [J]. Computer applications, 2003, 19(1): 6-9.

[6] Huang Wenmei et al. System Simulation Analysis —— MATLAB Language Engineering Application [M]. Changsha: University of Defense Science and Technology Press, 2001, 299-312.

[7] Yuan Huimei, Song Yu. Roman Digital Pattern Recognition Based on MATLAB Software [J]. Computer Applications and Software, 2007, 24(9): 143, 144, 183.