EasyChair Preprint

№ 3298

Calculating the Effect of Neural Network Parameters on Their Performance

Manhal Basher Hasan Aga

April 30, 2020

Calculating the effect of neural network parameters on their performance

Manhal Basher Hasan Aga
Northern Technical University, Mosul, Iraq. Email: manhalbasher@ntu.edu.iq & manhalbasher@gmail.com

Summary
In this research, a three-layer neuronal network was studied and designed, a network capable of learning a set of large data with the help of the reverse propagation error method, and studying the effect of parameter changes (learning step, number of nodes, type of activation dependent for a number of different income signals) and the severe impact This change in the work of the neural network causes it, as the results of these experiments demonstrated the extreme sensitivity of the designed neuron response, which relies on the propagation technique to change these parameters.

Key Words
Neural networks, back propagation algorithm, algorithm instruction, layers.

Introduction

The field of neuronal networks has evolved based on many factors such as neurobiology, mathematics, computer science, (artificial intelligence), physics, etc.

The modern theory of neuronal networks began since the 1940s in the work of Warren McCulloch and Walter Pitts, which demonstrated the possibility of neuronal networks in principle to perform a mathematical or logical operation called the new structure Perceptron. The work continued in the seventies by several researchers until a neuron network was developed that is able to function as a memory. In this article, I used the Back Propagation algorithm to train a Feed-Forward multi-layer neural network, so we take three layers of the neural network with a single input and output. The propagation algorithm is the most common algorithm now in the process of teaching neural networks.

The idea of this algorithm is based on adjusting weights so that the error between the actual output value and the desired output value decreases. This method requires calculating the error derivative with respect to weights, i.e. it requires knowing how to change the error with respect to its previous value, and adjusting the weights between the hidden layer and the output layer so that the error change rate in these weights is added to the previous weights, this process returns the layer to the income layer This gave the name the inverse propagation of this algorithm.

The importance of research and its goals

The aim of the research is to highlight the problems experienced by the current neuronal networks, and to study the effect of changing parameters (education step, number of nodes, type of activation dependent for a number of different income signals), and how to overcome the resulting errors based on the back propagation algorithm. The importance of the research lies in finding the ideal parameters for the neural network to obtain the least possible error.

Research method and materials

In this research, the following subjects were studied: -

- Building a neural network consisting of three layers with a single hidden layer, which is sufficient to represent the system (3).

The neural network training algorithm, which is the back propagation algorithm BP.

- Continued to estimate the performance of the neural network and it is a function of error.

The algorithm used in programming

Reverse propagation algorithm

We will study the multilayer neural network, employees of the propagation algorithm, to illustrate this process. We take three layers of the neural network with one input and one output as shown in Figure (1).

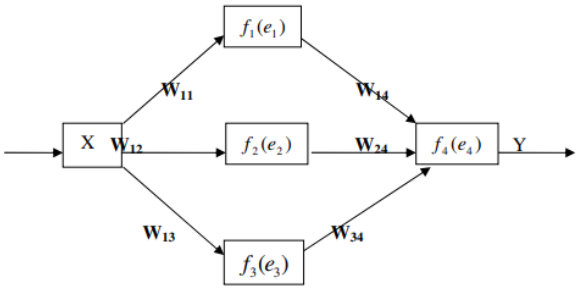

Figure (1) a multi-layer neuronal network

Each neuron is a combination of two units. The first unit produces weight and income signal coefficients. The second unit is that which invokes the neuronal activation function, where f () represents the activation function of each neuron, X also represents the input and Y the output, e represents the input group with its weights, and W represents the weights that connect the network neurons together with some entrances and exits .

To study the neural network, we need to train a set of data (X1, X2, X3, …… ..). With each exercise, the number of neural network weights is adjusted [4].

The modification is calculated using the algorithm described as follows:

Each education step begins with feeding the input signal from the training group. After this stage, we can determine the value of the output signals for each neuron in each layer of the network as shown in Figure (2).

Back Propagation

It is an algorithm for learning by example and it includes:

• In a multi-layered network, the information that nurtures the income contract can be represented internally so that the exits are generated from this internal representation [5].

• The output is determined by the internal operations of the network node.

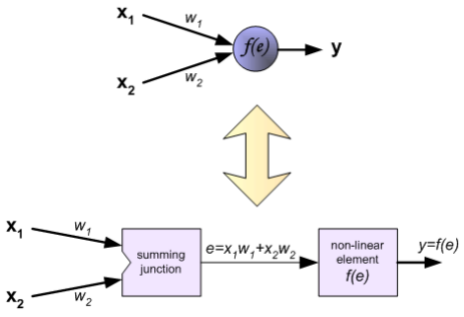

Figure (2) shows how to calculate the output from the input data and based on the neural network structure NN

The Learning Scenario

The network is led by a set of examples of income-output pairs (training set).

- Weights can be adjusted when the function of the dependent pairs is approximated.
- Then she checks the network through her ability to play the required role.

Correcting the error is:

1. During training, the income is placed in the network and flows through it, generating a set of values on the output units.
2. The actual output is calculated and then compared with the desired goal:

* If there is a match between the actual output and the target output, then there will be no change in the network.
* When there is no match, the change will take place.

Learning Algorithm

The teaching of the algorithm depends on the decline that gives us the error [6], as it expresses the error as follows:

$$\sum_{p} E_p$$

Where Ep is the error for one input, and E is the total error.

$$Ep = \frac{1}{2} \sum_i (ti - ai)^2$$

Where:

ai: is the actual output.

I : is the number of iterations.

We will make weights change based on the change in giving us the error

$$\Delta w = -\eta \nabla E$$

Where ŋ is the gradient constant (the education constant) determining the step size.

The change in weight that connects the neuron i to the neuron j is:

$$\nabla wji = - \eta \nabla jiE = \frac{\partial E}{\partial w_{ji}}$$

The ∂E / ∂wji term can be expressed depending on the structure of the neural network [7] NN.

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial a_j} = \frac{\partial a_j}{\partial net_j} = \frac{\partial net_j}{\partial w_{ji}}$$

The third partial derivative is calculated in equation (4) by specifying netj

net j = $\sum_{i=0}^{n} a_i * w_{ji}$

$$\frac{\partial net_j}{\partial w_{ji}} = \frac{\partial}{\partial w_{ji}} \sum_{k=0}^{n} w_{jk} a_k = \sum_{k=0}^{n} \frac{\partial w_{jk} a_k}{\partial w_{ji}}$$

By examining the partial derivative, we notice that (∂w) _jk / ∂ w _ji equals zero unless k = I

Therefore, equation (5) becomes:

$\frac{\partial net_j}{\partial w_{ji}}$ = ai

Returning to the partial derivative

Aj = f(netj) Since the $\frac{\partial a_j}{\partial net_j}$

f(netj) = $\frac{1}{1+e^{-netj}}$

f '(netj) = $\frac{d(1+e^{-x})^{-1}}{dnet_j} \frac{\partial a_j}{\partial net_j}$

$\frac{d(1+e^{-x})^{-1}}{dnet_j}$ = (-1)(1+e-x)-2 e-x (-1)

= $\frac{1+e^{-x}-1}{1+e^{-x}} \cdot \frac{1}{1+e^{-x}}$

= $\frac{1+e^{-x}-1}{1+e^{-x}} \cdot \frac{1}{1+e^{-x}} = \left(\frac{1+e^{-x}}{1+e^{-x}} - \frac{1}{1+e^{-x}}\right) \frac{1}{1+e^{-x}}$

= (1-aj)aj

So it is:

$\frac{\partial a_j}{\partial net_j}$ = = (1-aj)aj

To calculate the first derivative from equation (4): $\partial E/\partial a_j$, we take advantage of equation (2):

We will back to

EP = $\frac{1}{2}\sum_{i(} t_i - a_{i)}2$ (7)

Where the sum is on the output of the grid units, and we can study two states for the partial derivative

* J is the unit output
* J is for a hidden layer

If j is a unit output, then we can simply calculate the derivative as follows:

$\frac{\partial E}{\partial a_j} = \frac{\partial}{\partial a_j} \cdot \frac{1}{2}\sum_{i(} t_i a_{i)}2$

= $\sum_{i(} t_i - a_{i)} \frac{\partial(t_i - a_i)}{\partial a_j}$

= (ti-ai)(-1) = -(ti-ai)

∂j = - (ti-ai) (1-aj)aj

If aj is the hidden layer (h), then we need to rely on the gradient law applied to the units, k, to connect to the unit j [8].

$\frac{\partial E}{\partial w_{ji}}$ = .wij. f ' (neth).Xi (8)

f ' (neth) = $\frac{\partial out_h}{\partial net_h}$

wij = $\frac{\partial out_o}{\partial net_h}$

xi = $\frac{\partial net_h}{\partial w_i}$

The algorithm supported in reaching the appropriate network (network design):

After designing the network, it was trained where mistakes occurred, so we compared each error with the error immediately preceding it, so if the error contradicted, the training was good, and if it did not contradict; we increase the number of nodes and continue with this increase until the error decrease decreases as shown in Figure (3).

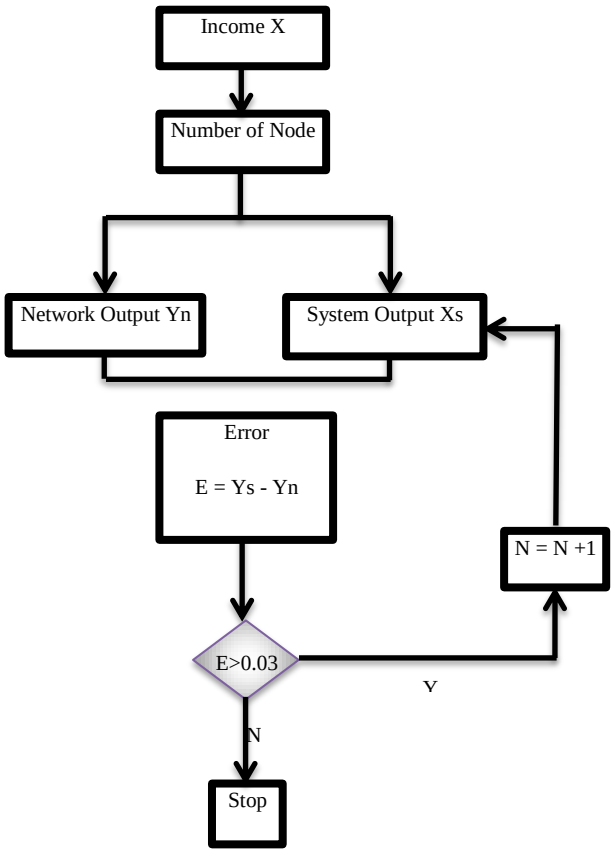

Figure (3) Show the algorithm used to access the appropriate network

Results and discussion:

In this paragraph, we discuss three main points:

1- Change Education Step (STEP_W).

2- Change of contract number (NO_OF_NODE).

3- Different income signals.

First: change the education step (STEP_W).

Where the sin (2πt / 250) function is the function that is used as network logon.

Table (1): The different effects of changing the education step on the performance of the neuronal network.

Number of times executed	Maximum number of nodes	Step education	Amount of error	The number of the contract after implementation	Total square error (After training)
1000	200	0.01	-0.002718	200	16.06
1000	200	0.001	-0.317749	200	188.83
1000	200	0.06	-0.000080	120	7.20

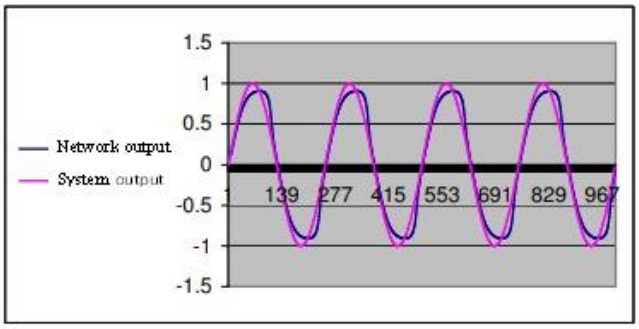

Figure (4) shows the convergence when the education step = 0.01

Number of repeats = 1000

The number of the maximum number = 200

Education step = 0.01

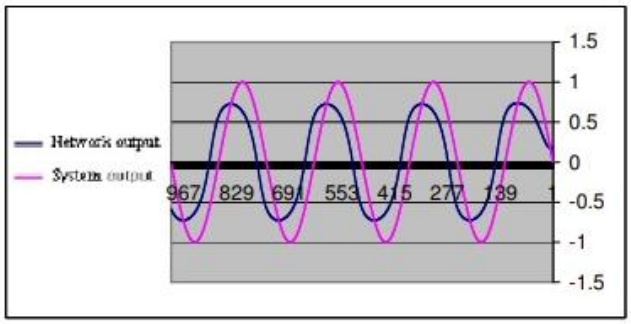

Figure (5) The effect of the learning step when it became = 0.001

Number of repeats = 1000

The number of the maximum number = 200

Education step = 0.001

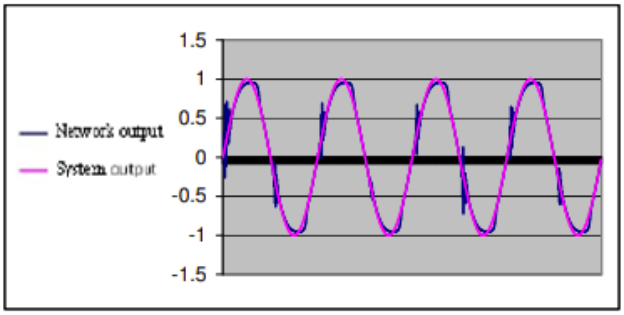

Figure (6) When education step = 0.06

Number of repeats = 1000

The number of the maximum number = 200

Education step = 0.06

Second: Change the number of nodes (NO_OF_NODE).

Where the sin (2πt / 250) function is the function that is used as network logon.

Table 2: Various effects of changing the number of nodes on the performance of the neuronal network.

Number of times executed	Maximum number of nodes	Step education	Amount of error	The number of the contract after implementation	Total square error (After training)
1000	50	0.06	- 0.001157	50	10.39
1000	100	0.06	- 0.000284	100	5.01
1000	150	0.06	- 0.000125	150	3.29

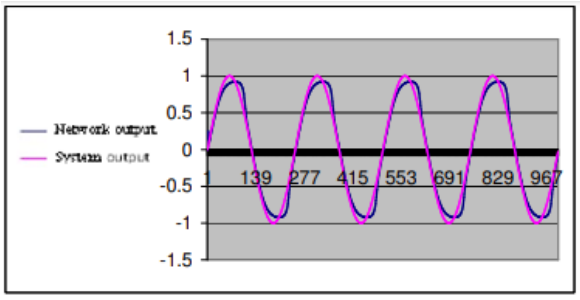

Figure (7) The network output when the number of nodes = 50

Number of repeats = 1000

Maximum number of nodes = 50

Education step = 0.06

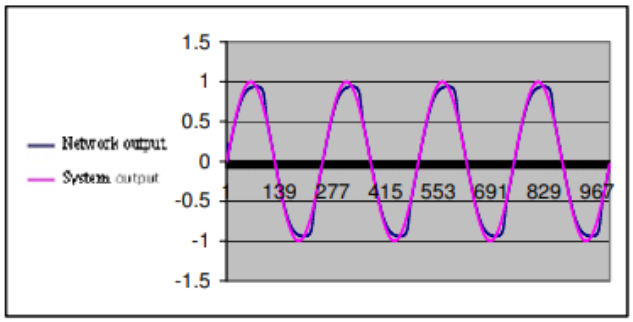

Figure (8) The network output when the number of nodes = 100

Number of repeats = 1000
The number of the maximum number = 100
Education step = 0.06

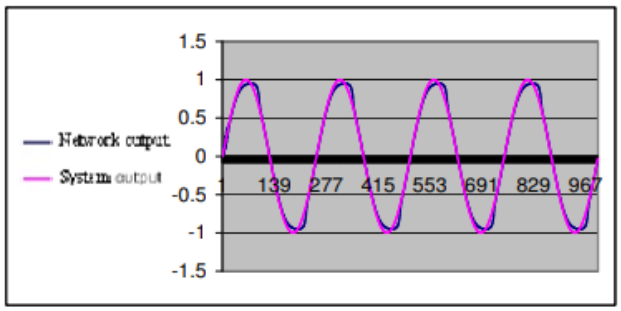

Figure (9) The network output when the number of nodes = 150

Number of repeats = 1000
The number of the maximum number = 150
Education step = 0.06

Type of input signal	Maximum number	Number of repetiti	Total square	Amount of error	Number of times	The number of the contract
SIN	200	1000	3.16	-0.000102	1063	200
COS	200	1000	375.78	-0.003784	1063	200
SIN2	150	2000	6.56	0.000001	2063	150
SIN*COS	200	1000	48.42	-0.306595	1063	115

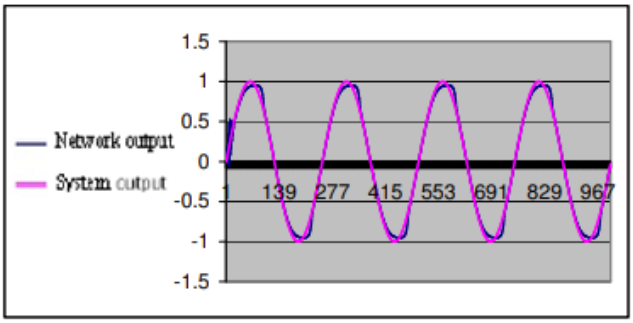

Figure (10) shows the network performance when the input signal was sin

Number of repeats = 1000
The number of the maximum number = 200
Education step = 0.06

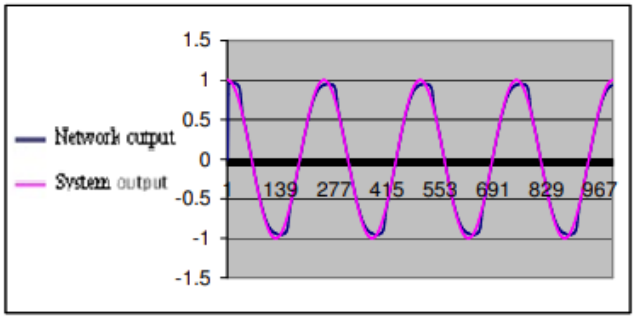

Figure (11) shows the network performance when the input signal was cos

Number of repeats = 1000
The number of the maximum number = 150
Education step = 0.05

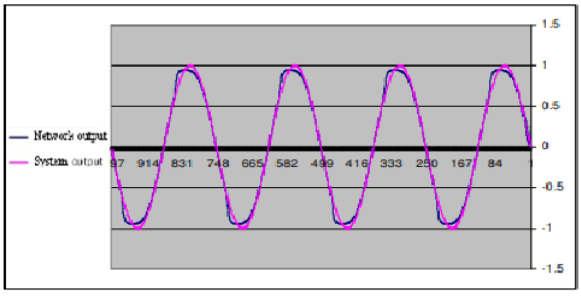

Figure (12) shows the network performance when the input signal was sin*cos

Number of repeats = 1000
The number of the maximum number = 200
The number of contracts after implementation = 115

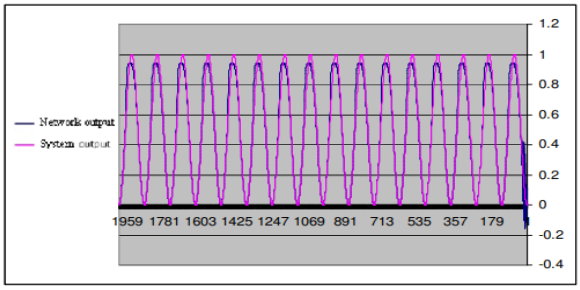

Figure (13) shows the network performance when the input signal was $\sin^2$

Number of repeats = 1000
The number of the maximum number = 150
Education step = 0.05

In Table (1), we find that the lower the step of instruction (STEP_W), the closer the actual output to the ideal output becomes worse, and this is illustrated by Figure (2), when when the teaching step was equal to 0.001, the sum of the square of the error was equal to 188.83, and the distance between the two graduates is greater What could.

As for table (2), it appears that the higher the number of nodes, the better the convergence, as in Figure 9 we find that the number of nodes (150) is the highest value in the table, and that the convergence between the two graduates is the best, where the value of the square of error (3.29) It is less than the rest of the values for the error square in Table (2), so convergence is better than this value.

In Table (3), we note that despite the income signal, the square function of the sinusoid is more complicated than its predecessors, but after the parameters were appropriately chosen, the convergence between the system output and the network output was good, as shown in Figure (13).

Conclusions and recommendations:

1- The results of the study showed that the posterior diffusion algorithm is a good way to train neuronal networks.
2- It is recommended to have a neuronal network structure consisting of one hidden layer.
3- The study showed that the selection of the SIGMOID subordinate was appropriate, because it enabled the network to approach the objective follower.
4- The results indicated the presence of defects using the proposed algorithm. Therefore, it is advised to use the genetic algorithm to reach the optimal results in the design of neuronal networks.

References:

1-
ERIC,D and PATRICK,N.", Neural Networks", university of Manchesters, Edition Eyrolles.Paris2010, 145P.
2-
SPRTORI,A.and ANTSAKLIS,P "Implements of learning control system using neural networks. "IEEE control systems magazine, April 2015,75.
3-
ADNAN,S.,TAYFUN,M.&SINAN,U.,"Determination of efficiency of flat-plate solar collectors using neural network approach",Ankara,Turkey,2008,1533-1539.
4-
MELLIT,A.,BENGHANEM,M.,&KALOGIROU, A.,"Modeling and simulation of a stand-alone photovoltaic system using an adaptive artificial neural network : proposition for anew sizing procedure",Renewable Energy,2007,285-313.
5- http://www.techguide.com.