International Refereed Journal of Engineering and Science (IRJES)
ISSN (Online) 2319-183X, (Print) 2319-1821
Volume 2, Issue 10 (October 2013), PP. 12-14

A study of neural network algorithms

Namrata Aneja
(M.C.A., M.Phill.), Dyal Singh College, Karnal., Haryana, India.

Abstract:- In this research paper I have tried to accumulate few different theories prevailing regarding neural networks algorithms. All the different theories have a different aspect to consider and have a different idea to work on. Though all have different process of gathering weights, different variables and parameters and different points , logics to establish, but one thing is common they all have used learning methods and have been successful in doing so.

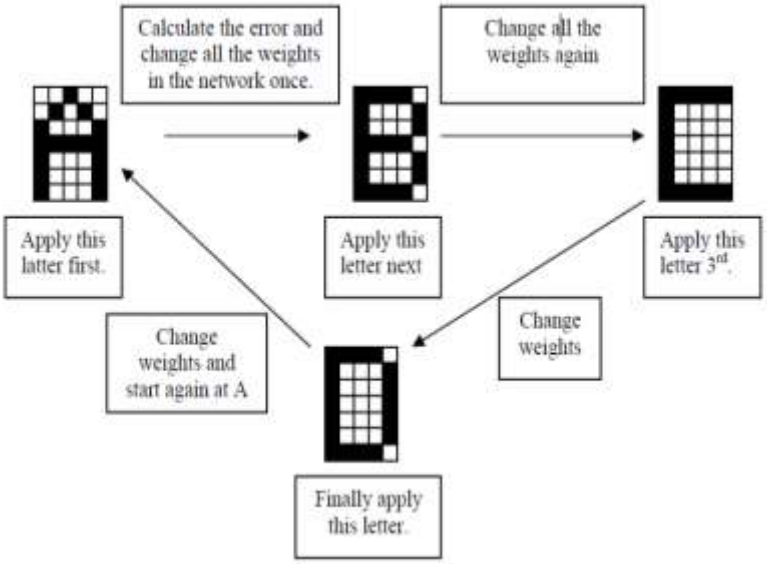
I. INTRODUCTION

The human brain is one of the most advanced forms of computer system. The neural network is controlled by the brain and through it; the brain is able to control various functions of the rest of the body. Due to this, computer scientists have undertaken deep study over the years of evolution of computing. This is the reason why several algorithms simulating the human neural networks have been invented. They are especially useful in the field of artificial intelligence.

II. BACK PROPAGATION ALGORITHM

Several algorithms have been developed in neural networks and one of them is the backward propagation algorithm. It is an intelligent network whose name is derived from its learning mechanism. It learns through experience/example. According to the requirements of the task, the algorithm alters the weights attached to the network. After it completes its training, it will output the results based on the original inputs. It is best suited for tasks involving recognition of patterns and mapping them onto actions. After the first few experiences, the algorithm learns and is able to conduct subsequent mapping independently.

The training of the algorithm begins by setting all the network weights to a small value (between -1 and 1). After application of the required input pattern, the results will be incorrect since the network weights are assigned randomly. To rectify this, the error is calculated by subtracting the output from what should have been the correct output. This is called supervised training where the input and expected output are already known. Based on the results of the calculation, the network weights are changed. The error keeps getting smaller with each error calculation and weight change until finally the results are acceptable. The figure below illustrates the learning process of a neural network in recognizing the first four letters of the alphabet.



From the example above, after several iterations of changing the network weights, the network should be able to correctly recognize the letters. At this stage, it is safe to stop the training since the network has been fully trained. However, the decision requires there be a set value of minimum error at which the training can stop. A validation set can also be used to stop the training since the network may slow down from overtraining

and being too correct. The backward propagation algorithm is very good at recognizing patterns. However, it lacks the capability to recognize patterns in a noisy environment with many distractions.

III. BACK PROPAGATION THROUGH TIME

The back propagation through time algorithm is another algorithm designed for the purpose of training networks. It is a variant of the back propagation algorithm that is used for recurrent networks training them through discrete time steps. Random numbers of feedback iterations may be included in the algorithm. However, the algorithm may lack connections in its inputs (Amir et al 2007).

This algorithm has several variations that have been widely used. The first is the back propagation through time with online update algorithm. Each weight's gradient is calculated and added to the back step copies of previous layers. The change in weights of the previous pattern is also factored in. The algorithm is especially useful in large networks since the updates of the patterns are made much more frequently.

Another variation of the back propagation through time algorithm is the batch propagation though time. The gradient for each weight is factored in by averaging it over the entire set undergoing the training.

Under the back propagation through time algorithm, the network processes pattern sequences through a series of defined activations. By applying an input pattern of zero, all the activities within the model can be set to zero from where processing begins.

The Levenberg-Marquardt algorithm

This algorithm is directly derived from numeric algebra. The main activity here is to solve the equation below.

$(J^t J + \lambda I)\bar{Q} = J^t E$

J stands for the Jacobian matrix calculated using the formula below

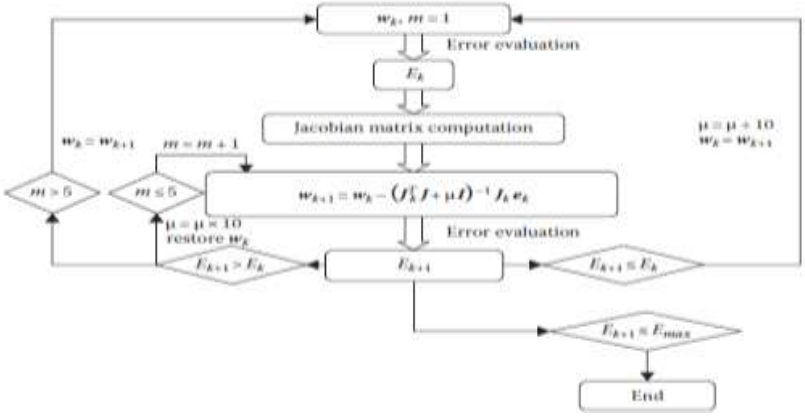
$$J = \begin{bmatrix} \frac{\partial F(x_1, w)}{\partial w_1} & \cdots & \frac{\partial F(x_1, w)}{\partial w_W} \\ \vdots & \ddots & \vdots \\ \frac{\partial F(x_N, w)}{\partial w_1} & \cdots & \frac{\partial F(x_N, w)}{\partial w_W} \end{bmatrix}.$$

It is an N by W matrix where N is the total number of inputs to the training and W the parameters of the training (weights). The chain rule of calculus can be very effective in calculating the Jacobian.

The $J^t J$ value is called the Hessian and can be accurately calculated if the errors within the function are minimized. λ is the damping factor in the equation and starts at a minimal value, as low as 0.1. After arriving at the solution to the equation, the $\bar{Q}$ value is added back to the weights to update them. The sum of squared errors is also calculated and added back to λ.

The algorithm is basically an iteration in which the value of λ is continually updated until the sum of squares of errors are minimal. The value of the initial network weights has a large impact on the working of this algorithm.

Though the calculation of the Jacobian matrix is a major part of this implementation, there is also the question of organizing the iterative training process. The algorithm works under the assumption that if the error is lower than the preceding one, then the quadratic approximation function is doing its work well. If it goes higher, this means that there is need for examination of the gradient of curvature and finding a new combination coefficient. The figure below summarizes the training process of this algorithm.



This algorithm is efficient in solving the problems associated with methods such as the Gauss-Newton and the gradient descent methods. It is therefore one of the most efficient training algorithms today.

IV. THE GENETIC ALGORITHM APPLIED TO THE HOPFIELD NETWORK

The Hopfield network is another neural network that has been developed. This network is made of a set of two neurons that have been fully connected. The two neurons have either an on or off status giving them their characteristic two state binary nature. The neurons take input and the total value of their inputs determines the state of the neuron. This network cannot be trained using all the training algorithms available since tests have shown that it can only be trained using six of the available patterns.

The speed of training the Hopfield network is one epoch. This is enough to train the entire network in two iterations. The size of the network is 14520 units. the Hopfield network uses a basic model in which all the neurons have been connected together. The weights on the connections between neurons are systematically arranged using an organized matrix of w_{ij}. The learning process of this network begins by setting the nodes to some known value (Karayiannis et al 1993). Several iterations are then performed through either asynchronous or synchronous update methods. This continues for a while and then stops at which point the neurons are read and the network formed interpreted. The model works on the assumption that networks are stored in the weight matrices.

The input to the algorithm should contain an embedded pattern. Using a mechanism of auto-association, the network can divide the patterns stored in the network into two parts; a cue and the association. To retrieve the network from the matrix, the user simply enters the cue. The model offers two options for a weight matrix, being +1 or -1. For a value of 1, it could have the following two matrices (1, 1) or (-1, -1). Both of these are stable states. For a value of -1, the matrices are either (1, -1) or (-1, 1). The states are updated synchronously and therefore may change accordingly.

This model is heavily dependent on the values of its weight matrix since this is where the patterns are derived from. Therefore, understanding this model begins by first understanding the mechanism of setting the weight matrices. If the weight between two neurons has a positive value, they will exhibit the tendency to drag each other in the same direction and vice versa (Patrick 2008).

The Hopfield network is based on the recall mechanism of the human brain. This is called content addressable memory. The input values to the model contain an embedded pattern that is then interpreted by the model's mechanisms. In the event that some of the neurons are destroyed, the network's performance degrades. However, this does not mean that it loses all its functionality. Just like in the human brain where brain damage does not mean the total loss of all functionality, some of the network's functionality still remains even with severe network damage (Alexander et al 2011).

V. CONCLUSION

In conclusion it can be said that back propogation algorithm is efficiently effective in pattern recognition where as the levenberg and Marquart are used for guassian newton and gradient metods. Where as Hopfield network is an emulation of working of neuron used in recalling process in brain, but all the algorithm have been successful in proving the fact of learning. Which is done by all the networks discussed.

WORKS CITED

[1]. Alexander g, erik, ivan, róbert s, martina v. *Comparison of neural networks learning algorithms for simulation of rectilinear snake locomotion*. Modelling of mechanical and mechatronic systems, 2011.

[2]. Amir A S, Mohammad B T, and Abbas H. *Modified Levenberg-Marquardt Method for Neural Networks Training.* World Academy of Science, Engineering and Technology 6, 2007.

[3]. Haykin, Simon S. *Neural Networks: A Comprehensive Foundation*. New York: Macmillan, 1994. Print.

[4]. Karayiannis, N B, and A N. Venetsanopoulos. *Artificial Neural Networks: Learning Algorithms, Performance Evaluation, and Applications*. Boston: Kluwer Academic, 1993. Print.

[5]. MacKay, David J. C. *Information Theory, Inference, and Learning Algorithms*. Cambridge, UK: Cambridge University Press, 2003. Print.

[6]. Mandic, Danilo P, and Jonathon A. Chambers. *Recurrent Neural Networks for Prediction: Learning Algorithms, Architectures, and Stability*. Chichester: John Wiley, 2001.

[7]. P. Patrick van &r Smagt. A comparative study of neural network algorithms applied to optical character recognition, 2008.