Impact of Mutation Weights on Training Backpropagation Neural Networks

Lamia AbedNoor Muhammed
College of Computer and Mathematics Sciences
University of Al-Qadissiyah
Iraq

Abstract

Neural network is a computational approach, which based on the . This approach is conducted by several parameters; learning rate, initialized weights, network architecture, and so on, However, in this paper would be focus on one of these parameters that is weights. The aim is to shed lights on the mutation weights through training the network and its effects on the results. The experiment was done using backpropagation neural network with one hidden layer . The results reveal the role of mutation in escape from the local minima and making the change.

1.Introduction

The back-propagation algorithm has been the most popular and most widely implemented for training these types of neural network. When using the back-propagation algorithm to train a multilayer neural network, the designer is required to arbitrarily select parameter such as the network topology, initial weights and biases, a learning rate value, the activation function, and a value for the gain in the activation function. An improper choice of any of these parameters can result in slow convergence or even network paralysis where the training process comes to a virtual standstill. Another problem is the tendency of the steepest descent technique, which is used in the training process, to get stuck at local minima[1]. There is more work to deals with these problem. The tangent plane algorithm is a fast sequential learning method for multilayered feedforward neural networks that accepts almost zero initial conditions for the connection weights with the expectation that only the minimum number of weights will be activated. However, the inclusion of a tendency to move away from the origin in weight space can lead to large weights that are harmful to generalization. This paper evaluates two techniques used to limit the size of the weights, weight growing and weight elimination, in the tangent plane algorithm. Comparative tests were carried out using the Extreme Learning Machine which is a fast global minimiser giving good generalization. Experimental results show that the generalization performance of the tangent plane algorithm with weight elimination is at least as good as the ELM algorithm making it a suitable alternative for problems that involve time varying data such as EEG and ECG signals[2].

The method of the graphical interpretation of the single-layer network weights is introduced. It is shown that the network parameters can be converted to the image and their particular elements are the pixels. For this purpose, weight-to-pixel conversion formula is used. Moreover, new weights' modification method is proposed. The weight coefficients are computed on the basis of pixel values for which image filtration algorithms are implemented. The approach is applied to the weights of three types of the models: single-layer network, two-layer backpropagation network and the hybrid network. The performance of the models is then compared on two independent data sets. By means of the experiments, it is presented that the adjustment of the weights to new values decreases test error value compared to the error obtained for initial set of weights[3].

A novel weight initialization method for the random neural network was proposed[4]. The method relies on approximating the signal-flow equations of the network to obtain a linear system of equations with nonnegativity constraints. For the solution of the formulated linear nonnegative least squares problem we have developed a projected gradient algorithm. It is shown that supervised learning with the developed

initialization method has better performance in terms of both solution quality and execution time than learning with random initialization when applied to a combinatorial optimization emergency response problem[4].

The aim of this paper is to present a method for training ANN. The ability of metaheuristics and greedy gradient based algorithms are combined to obtain a hybrid improved opposition based particle swarm optimization and a back propagation algorithm with the momentum term. Opposition based learning and random perturbation help population diversification during the iteration. Use of time-varying parameter improves the search ability of standard PSO, and constriction factor guarantees particles convergence. Since several contingent local minima conditions may happen in the weight space, a new cross validation method is proposed to prevent over fitting[5].

In the proposed work[6], the optimization of architectures and connection weights uses the evolutionary process. A single-point crossover is applied with selective schemas on the network space and evolution is introduced in the mutation stage so that an optimized ANNs are achieved[6].

2.Artificial neural networks (ANN)

Artificial neural networks (ANN) have been developed as generalizations of mathematical models of biological nervous systems. A first wave of interest in neural networks (also known as *connectionist models* or *parallel distributed processing*) emerged after the introduction of simplified neurons by McCulloch and Pitts . The basic processing elements of neural networks are called *artificial neurons*, or *simply neurons* or *nodes*. In a simplified mathematical model of the neuron, the effects of the synapses are represented by connection weights that modulate the effect of the associated input signals, and the nonlinear characteristic exhibited by neurons is represented by a transfer function. The neuron impulse is then computed as the weighted sum of the input signals, transformed by the transfer function. The learning capability of an artificial neuron is achieved by adjusting the weights in accordance to the chosen learning algorithm[7]. Artificial neural networks are classified according to the architecture into different classes. Multilayer FeedForward Network distinguishes itself by the presence one or more layers, whose computation nodes are correspondingly called hidden neurons or hidden units[8].

The most popular class of multilayer feed-forward networks is multilayer perceptrons in which each computational unit employs either the thresholding function or the sigmoid function. Multilayer perceptrons can form arbitrarily complex decision boundaries and represent any Boolean function[9].

3.Backpropagation algorithm

Backpropagation algorithm a popular method for the training of multilayer perceptrons . The training proceeds in two phases:

1- In the forward phase, the synaptic weights of the network are fixed and the input signal is propagated through the network-layer by layer, until it reaches the output. Thus, in this phase, changes are confined to the activation potentials and outputs of the neurons in the network.

2- In the backward phase, an error signal is produced by comparing the output of the network with a desired response. The resulting error signal is propagated through the network, again layer by layer, but this time the propagation is performed in the backward direction. In this second phase, successive adjustments are made to the synaptic weights of the network. Calculation of the adjustments for the output layer is straightforward, but it is more challenging for the hidden layers[8].

4.Experiments and Results

The practical work was done through construct feedforward neural network with one hidden layer contains two hidden neurons. Then training this network with backpropagation algorithm. The data used in this experiment is "iris data" that found in the UCI repository data for machine learning experiment. The problem was introduced with this data to network is a classification, where the instances in this data are labeled with one of three classes. The parameters were used in training process are shown in table(1). These parameters were unchanged during the experiment in order to realize the aim of this paper.

Table(1) specification of parameters used in experiment work

Parameters	Specification
no. of hidden layers	1
no. of hidden neurons	2
no. of input neurons	4
no. of output neurons	1
learning rate	0.005
no. of epoch	10000
type of output data	integer data
type of input data	continuous data

Then the following steps were applied:

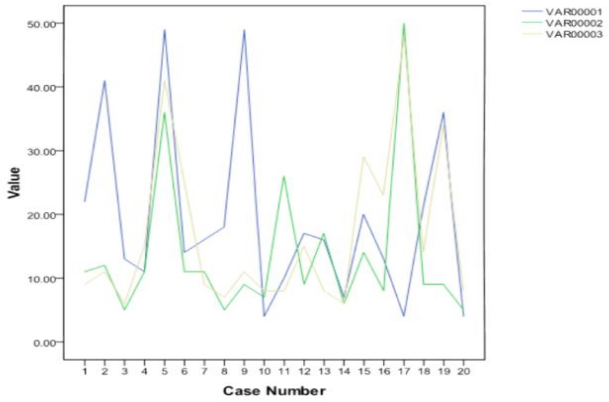
a- Training the network without any mutation and calculate the absolute error for training.
b- Training the network with the same initial weights that were used in step(a) and mutation random individual weights through training process when the error didn't change for long time. Then compute the absolute error at the end. This step was executed for two rounds (first round and second round) in order to confirm the results effectively as shown in table(2).

Table(2) Global absolute error from training neural network

No. of run	Without mutation	First round with mutation	Second round with mutation
1	22	11	9
2	41	12	11
3	13	5	6
4	11	11	15
5	49	36	41
6	14	11	25
7	16	11	9
8	18	5	7
9	49	9	11
10	4	7	8
11	10	26	8
12	17	9	15
13	16	17	8
14	7	6	6
15	20	14	29
16	13	8	23
17	4	50	48
18	21	9	14
19	22	11	9
20	41	12	11

To distinguish the results, figure(1) explain graphically the disparity of the errors that produced. Three plot lines represent errors for 20 case in three groups of runs.

Figure(1) global errors from



The results reveal several facts. In case(1,2,3,7,8,9,18,20), the error in run dropped. This means, the error come from worst weights that were mutated. In case (17) , the error in run without mutation was low, but when the mutation was executed , the error increase. That means the mutation incurred the good weights. In case(14,5), good or worst error value in run without mutation didn't change in the runs with mutation. That means, the mutated weights were negligible.

5.Conclusion

From the previous review, we appoint to

1- The good weights of neural network with backpropagation may be lost when the whole weights are displaced with new ones .
2- The individuals weights can be changed in order to improve the results with specific restriction that is mutation randomly in greedy way in order to locate the worst weights and then change them.

3- The mutation can improve training especially when the error of network still unchanged with high value

Reference

[1] N. Nazri Mohd, R.S. Ransing, A. Norhamreeza," BPGD-AG: A New Improvement Of Back-Propagation Neural Network Learning Algorithms With Adaptive Gain",Journal of Science and Technology, Vol 2, No 2 (2010), p.p.83-102.

[2] P. May, E. Zhou, C. W. Lee, " A Comprehensive Evaluation of Weight Growth and Weight Elimination Methods Using the Tangent Plane Algorithm, (IJACSA) International Journal of Advanced Computer Science and Applications", Vol. 4, No. 6, 2013, p.p. 149-156.

[3] K. Maciej and S. Damian, "Influence of graphical weights' interpretation and filtration algorithms on generalization ability of neural networks applied to digit recognition", Neural Comput & Applic (2012) 21:1783–1790.

[4] T. Stelios, "A novel weight initialization method for the random neural network", Neurocomputing 73 (2009) 160–168

[5] Y. Masoud, K. Mohammad M. and ,F. Mehdi, "A hybrid algorithm for artificial neural network training", Engineering Applications of Artificial Intelligence 26 (2013) 293–301

[6] G. V. R. Sagar, and S. Venkata Chalam, " Simultaneous Evolution of Architecture and Connection Weights in Artificial Neural Network", International Journal of Computer Applications (0975 – 8887) Volume 53– No.4, September 2012, p.p. 23-28.

[7] A. Ajith, "Artificial Neural Networks", *Handbook of Measuring System Design*, edited by Peter H. Sydenham and Richard Thorn.(c) 2005 John Wiley & Sons, Ltd, p.p. 901-908.

[8] H. Simon, 2010, "Neural Networks and Learning Machines", third edition, PHI Learning Private Limited New Delhi, p.p.123-124.

[9] J. Ani1 K. and M. Jianchang, "Artificial Neural Networks: A tutorial", 1996 IEEE, March 1996, p.p.31-41.