Applied Soft Computing 11 (2011) 4458–4464

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Review article

A self-organizing map-based initialization for hybrid training of feedforward neural networks

Mounir Ben Nasr*, Mohamed Chtourou

Research Unit on Intelligent Control, Design & Optimization of Complex Systems (ICOS), Department of Electrical Engineering, ENIS, BP W, 3038 Sfax, Tunisia

ARTICLE INFO

Article history:
Received 17 January 2009
Received in revised form 2 June 2010
Accepted 2 May 2011
Available online 7 May 2011

Keywords:
Constructive algorithm
Feedforward neural network
Gradient descent method
Supervised and unsupervised learning
Self-organizing map
Hybrid learning

ABSTRACT

This paper presents a novel hybrid algorithm for feedforward neural networks, called a self organizing map-based initialization for hybrid training based on a two stage learning approach. First stage, a structure learning scheme which includes adding hidden neurons is used to determine the network size. Second stage, a FN (fuzzy neighborhood)-based hybrid learning scheme which we have recently proposed is used to adjust the network parameters. In this approach the weights between input and hidden layers are firstly adjusted by Kohonen algorithm with fuzzy neighborhood, whereas the weights connecting hidden and output layers are adjusted using gradient descent method. Four simulation examples are provided to demonstrate the efficiency of the approach compared with other well-known and recently proposed learning methods.

Contents

1. Introduction

The backpropagation (BP) algorithm [1] has been greatly used for the supervised training of feedforward neural networks (FNNs). However, as it is well-known, this method has a slow convergence. Several methods have been proposed to speed up this method, such as momentum [1,19], adaptive learning rate [2,3,22,23], stochastic learning [4], recursive least square methods [13–15], regularization parameter [7,24–26], and statistical characteristics of the input pattern [45,46]. More significant improvement was possible by using various second order algorithms [5,8]. Many other algorithms with the emphasis on hybrid techniques have been developed to accelerate the training method of feedforward

* Corresponding author. Tel.: +216 4 274 088; fax: +216 4 275 595.
E-mail addresses: Nasr_mounir@yahoo.fr (M.B. Nasr), Mohamed.Chtourou@enis.rnu.tn (M. Chtourou).

1568-4946/$ – see front matter
doi:10.1016/j.asoc.2011.05.017

neural network [9,10,17,18,20,27]. Although neural networks are universal approximators [40] with sufficient hidden neurons, how to select an appropriate network structure is still an open problem. As a consequence, some techniques have been proposed to provide structure to the hidden layer of feedforward neural networks in order to increase learning speed and improve generalization accuracy [30,37–39,44]. To obtain the network structure automatically, constructive and destructive algorithms can be used. The constructive algorithms start with a small network and then grow additional hidden neurons and weights until a satisfactory solution is found [31–34]. The destructive algorithm starts with a large network and removes unnecessary network parameters [35,36]. The purpose of this paper is to apply a new approach, which is motivated by the work of the authors [17,20], to solve the problem of tuning both network structure and parameters of a feedforward neural network. In [17], we have developed a hybrid training algorithm for FNN which combines unsupervised and supervised learning. In this approach, the weights between input and hidden layers were determined according to an unsupervised procedure relying on the Kohonen algorithm with a bubble neighborhood function and the weights between hidden and output layers were updated according to a supervised procedure based on gradient descent method. An improved version of this technique called a FN (fuzzy neighborhood)-based hybrid has been also recently proposed [20]. This learning approach uses a Kohonen algorithm with fuzzy neighborhood for clustering the weights of the hidden layer and gradient descent method for training the weights of the output layer.

Both these approaches [17,20] suffer from the drawback posed by the fixed structure of the classical self organizing map (SOM). When using the SOM, the size of the grid and the number of neurons have to be predetermined. The need for predetermining the structure of the networks results in a significant limitation on the final mapping. In order to solve the limitation of static structure of SOM, many structurally adaptive self-organizing networks were proposed [41,42].

In this paper, we propose a new alternative hybrid algorithm for training a FNN, namely a self organizing map-based initialization for hybrid training. This algorithm could conceptually be split-up into two stages. In the first stage, a structure learning which includes adding hidden neurons is used to determine the network size. This learning algorithm starts with a single training pattern and a single hidden layer neuron. When a new pattern is presented to the network, the distance between the input pattern and the weight vector is computed. If the distance is smaller than a pre-specified distance threshold, the new input pattern belongs to this neuron, otherwise, this new pattern initiates a new neuron of hidden layer. This procedure is repeated until all patterns are presented. The network can automatically create the neurons of the hidden layer and their initial weights. This approach for constructing the self organizing map is proposed as a dynamic version of the Kohonen self organizing map in order to overcome the weakness of the need for user defined static map structure of SOM. In the second stage, network parameters are adjusted using a recently proposed approach [20]. This learning procedure uses different learning algorithms separately. The weights between input and hidden layers are firstly adjusted by a self organized learning procedure [6], whereas the weights connecting hidden and output layers are trained by a supervised learning algorithm, such as a gradient descent method [1].

The paper is organized as follows: In Section 2, we present the general principles of the proposed learning method. In Section 3, simulation results and comparisons with most commonly used learning algorithms are given. Finally, in Section 4, we present the main conclusions.

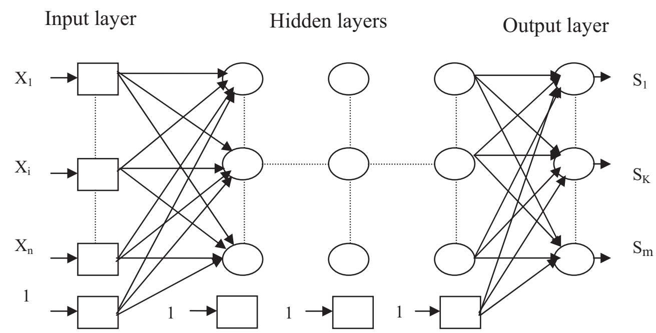

Fig. 1. Feedforward neural network.

2. Description of the learning algorithm

2.1. Preliminary

Feedforward neural network (FNN) is composed of one or more layers, between the output layer and the input layer. In layered architectures all neurons from one layer are connected to all other neurons in the following layer. Fig. 1 shows the general structure of the layered architecture.

Let n be the number of neurons in the input layer, m the number of neurons in the output layer, N_l the number of neurons belonging to the lth layer, and $o_k^{(l)}$ be the output of the kth neuron of the lth layer. The computation performed by each neuron can be expressed as:

$$net_k^{(l)} = \sum_{j=1}^{N_{l-1}} w_{kj}^{(l)} . o_j^{(l-1)} \tag{1}$$

$$o_k^{(l)} = f(net_k^{(l)} - \theta) \tag{2}$$

where $net_k^{(l)}$ is the weighted sum of neurons of the $(l-1)$th layer, $w_{kj}^{(l)}$ is the weight by which the same neuron multiplies the output $o_j^{(l-1)}$ of the jth neuron of the previous layer, θ is the threshold and $f(.)$ is a non-linear bounded function, often the sigmoid function. As shown in Fig. 1, the input signal is processed and relayed from one layer to the other, until the final result has been computed.

The BP algorithm has become the standard algorithm used for training FNN. It is a generalized least mean squared algorithm that minimizes a criterion which equals to the sum of the squares of the errors between the actual and the desired outputs. This criterion is

$$E = \frac{1}{2} \sum_{p=1}^{nb} ||d_p - s_p||^2 \tag{3}$$

where nb is the number of training vectors and d_p and s_p are the desired and the current outputs of the network for pth training vector.

The FNN with at least one hidden layer has the capability to approximate any desired non-linear function to an arbitrary degree of accuracy [16].

2.2. Training algorithm

We consider a FNN with three layers (Fig. 2): an input layer, a hidden layer and an output layer. The intermediate layer is considered like a self organizing map of Kohonen [6]. The number of neurons in the input and output layers is the same as the number of inputs and outputs of the problem.