Neurocomputing 182 (2016) 154–164

Contents lists available at ScienceDirect

Neurocomputing

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

# Constructive algorithm for fully connected cascade feedforward neural networks

Junfei Qiao [a,c,*], Fanjun Li [a,b,c], Honggui Han [a,c], Wenjing Li [a,c]

[a] College of Electronic Information and Control Engineering, Beijing University of Technology, Beijing 100124, China
[b] School of Mathematical Science, University of Jinan, Shandong 250022, China
[c] Beijing Key Laboratory of Computational Intelligence and Intelligent System, Beijing 100124, China

ARTICLE INFO

*Article history:*
Received 16 May 2014
Received in revised form
28 October 2015
Accepted 8 December 2015
Communicated by M.T. Manry
Available online 17 December 2015

*Keywords:*
Constructive algorithm
Feedforward neural network
Cascade correlation network
Convergence
Orthogonal least squares

ABSTRACT

In this paper, a novel constructive algorithm, named fast cascade neural network (FCNN), is proposed to design the fully connected cascade feedforward neural network (FCCFNN). First, a modified index, based on the orthogonal least square method, is derived to select new hidden units from candidate pools. Each hidden unit leads to the maximal reduction of the sum of squared errors. Secondly, the input weights and biases of hidden units are randomly generated and remain unchanged during the learning process. The weights, which connect the input and hidden units with the output units, are calculated after all necessary units have been added. Thirdly, the convergence of FCNN is guaranteed in theory. Finally, the performance of FCNN is evaluated on some artificial and real-world benchmark problems. Simulation results show that the proposed FCNN algorithm has better generalization performance and faster learning speed than some existing algorithms.



## 1. Introduction

Feedforward neural networks (FNNs) have been extensively used in many practical areas, such as controlling, modeling, image processing and so on [1–4]. The performance of FNNs is greatly dependent on their architectures [5]. Due to its structural flexibility, effective training algorithms and universal approximation ability, the architecture of multilayer perceptron (MLP) has been widely used during the past several decades [1–3,5,6]. While, recent research results have shown that the fully connected cascade feedforward neural network (FCCFNN) is more powerful than both MLP and bridged multilayer perceptron (BMLP) in most cases [7,8]. Unfortunately, it is difficult for FCCFNNs to decide the architectures and the training algorithms which are used to tune their weights and biases. Therefore, this paper will focus on constructing and training the FCCFNNs to get fast learning speed and good generalization performance.

The cascade correlation (CC) algorithm [9], one of the most popular constructive algorithms, can be used to construct a special FCCFNN, called cascade correlation network (CCN) [10–12]. The CC algorithm is simple and fast, since only the new hidden unit (a candidate unit) is trained before being added to the existent networks, and the input weights of the hidden units are frozen in the later process. Moreover, the CC algorithm determines the CCN's size, topology and weights simultaneously [13]. However, the weights connected to the output units are trained repeatedly after each new hidden unit is added, which usually causes heavy computation. In addition, the objective function used to train the new hidden units can not guarantee a maximal error reduction when a new hidden unit is added, which may lead to a large network with poor generalization performance. To solve these problems, orthogonal least squares based cascade network (OLSCN) algorithm was proposed [14]. Based on the orthogonal least squares (OLS) method, the OLSCN algorithm derives a new objective function to train the new hidden units. The weights connected with the output unit are updated after all necessary hidden units are added. However, when the candidate unit is linearly dependent with the existing hidden units, the new objective function may make mistakes. In addition, the linearly independence of the input vectors is necessary for the QR factorization [15], while the OLSCN algorithm can not guaranteed it. Especially, the OLSCN algorithm updates the weights of candidate hidden units by a modified Newton's method with gradient vector and Hessian matrix, which may result in local minimum, slow convergence and heavy calculation for larger networks.

Some gradient-based algorithms for training FCCFNNs have already been developed, such as error back propagation (EBP), Levenberg–Marquardt (LM) and neuron-by-neuron (NBN) algorithms [16–18]. Although all of them have succeeded in many

* Corresponding author at: College of Electronic Information and Control Engineering, Beijing University of Technology, Beijing 100124, China.
*E-mail address:* isibox@sina.com (J. Qiao).

http://dx.doi.org/10.1016/j.neucom.2015.12.003
0925-2312/

cases, they still have some connatural defects such as poor convergence and local minimum. To solve these problems, the training algorithms based on random hidden units have been proposed to train feedforward neural networks, such as the random vector version of the functional-link (RVFL) net [19,20], the extreme learning machine (ELM) for single-hidden layer feedforward neural network (SLFN) [21,22] and the No-Prop algorithm for multilayer neural networks [23]. These algorithms randomly choose the input weights of hidden units and analytically determine the output weights by simple linear regression. In this paper, the algorithms with random hidden units are described as random mapping (RM) algorithms. The RM algorithms have many advantages over traditional gradient-based learning algorithms, such as extremely fast learning speed, favorable generalization performance and simple codes. They have been applied to various problems [24–26]. However, there are few homologous algorithms to construct or train the FCCFNNs.

In this work, a novel constructive algorithm named faster cascade neural network (FCNN) is proposed to construct and train FCCFNNs automatically. The aim is to improve the generalization performance of CCN and OLSCN with fast learning speed. As shown in Fig. 1, the FCNN begins with an empty network without input and hidden units. The modified index is derived by the OLS method to select the new hidden units from the candidate pools. The candidate hidden units for FCNN are generated randomly inspired by RM algorithm, which means that the input weights and biases of candidate hidden units are chosen randomly according to some probability distribution and remain unchanged. The process of FCNN algorithm can be split into three parts. First of all, the linearly independent variable is selected by Gram–Schmidt orthogonalization method and added to the network one by one. Then, the candidate unit which causes the maximal reduction of the sum of squared errors is added to the existent network. After all the necessary hidden units have been added, the FCCFNN can be simply considered as a linear system, and the output weights are analytically determined by back substitution method. In a word, the proposed FCNN algorithm has several contributions as follows.

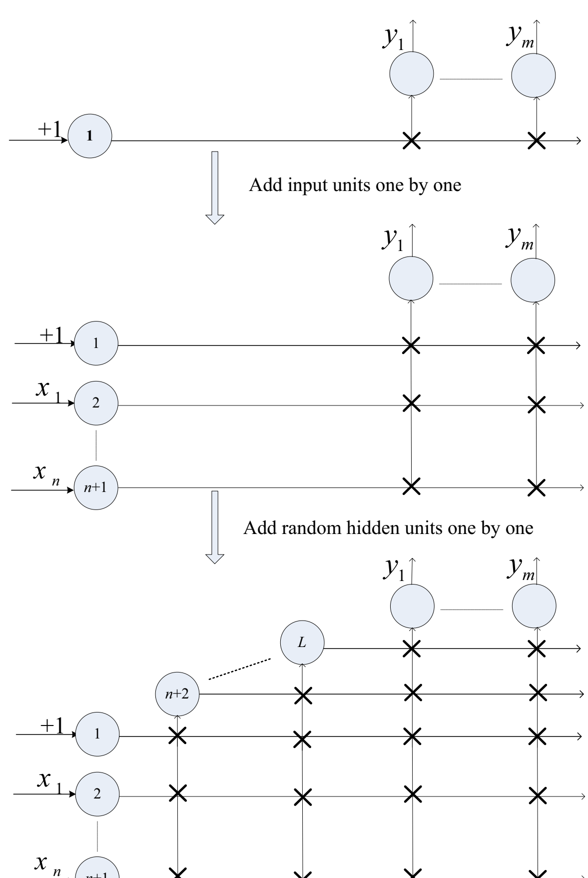


**Fig. 1.** The architecture of FCNN.

(1) A modified index is proposed, based on the OLS, to select new hidden units with maximal error reductions from candidate pools.
(2) A simple training method for weights is integrated into the proposed algorithm. No weights need to be tuned until all necessary hidden units are added. Only the output weights are calculated by back substitution while the others retain constant.
(3) A novel algorithm is proposed to automatically construct train FCCFNNs with fast learning speed and good generalization performance. Moreover, the convergence of the proposed FCNN algorithm is guaranteed in theory.

This paper is organized into six sections. Section 2 briefly introduces the preliminary (architecture, modified index and method to train the weights) for FCNN. The steps and convergence of the proposed FCNN algorithm are described in Section 3. Section 4 presents the experiment results that show the superior performance of FCNN compared with the original CCN and OLSCN algorithms. Some discussions about the generalization performance of FCNN are given in Section 5. Finally, Section 6 summarizes the main conclusions.

## 2. Preliminary

### 2.1. Architecture of FCNN and its mathematical expression

The architecture of FCNN is similar to that of FCCFNN and CCN. As shown in Fig. 1, all hidden units connect with each other and connections can be across all the layers. The difference is that the architecture of FCNN begin with an empty network without input and hidden units, which means that the input units and hidden units are added to the network one by one, while FCCFNN is a network with fixed topology. It can be seen from Fig. 1 that, different from the architecture of MLP and BMLP, each hidden unit in FCNN forms one layer, which receives connections from the network's original inputs and the pre-existing hidden units as well. Meanwhile, every original input as well as the output of each added hidden unit is connected to every output unit by a connection. Also, there is a bias input connected with all hidden units and output units, which is permanently set to +1. In the next section, we will present the architecture of FCNN by its mathematical expression.

Consider $N$ arbitrary distinct samples $\{(\boldsymbol{x}_i, \boldsymbol{t}_i) | \boldsymbol{x}_i \in R^n, \boldsymbol{t}_i \in R^m, 1 \le i \le N\}$, where $\boldsymbol{x}_i = (x_{i1} \quad x_{i2} \quad \cdots \quad x_{in})^{\mathrm{T}}$, $\boldsymbol{t}_i \in (t_{i1}, \ t_{i2}, \ \cdots, \ t_{im})^{\mathrm{T}}$. For simplifying formulation, we number the bias unit, input units and hidden units in sequence as done in [14], i.e., the $i$th unit defined as follows,

$$\text{the } i\text{th unit} = \begin{cases} \text{bias unit} & i = 1 \\ \text{the } (i-1)\text{th input unit} & 2 \le i \le n+1 \\ \text{the } (i-n-1)\text{th hidden unit} & n+1 < i \le L \end{cases}. \tag{1}$$

The input weights of the $k$th unit are $\{w_{lk} | l = 1, 2, \cdots, k-1\}$ where $w_{1k}$ is the bias of the $k$th unit and $k > n+1$. With the above definitions, the outputs of bias unit, input units and hidden units