Available online at www.sciencedirect.com
ScienceDirect
Neurocomputing 70 (2006) 489–501
NEUROCOMPUTING
www.elsevier.com/locate/neucom

Extreme learning machine: Theory and applications

Guang-Bin Huang*, Qin-Yu Zhu, Chee-Kheong Siew

School of Electrical and Electronic Engineering, NanyangTechnological University, Nanyang Avenue, Singapore 639798, Singapore

Received 27 January 2005; received in revised form 1 December 2005; accepted 3 December 2005
Communicated by J. Tin-Yau Kwok
Available online 16 May 2006

Abstract

It is clear that the learning speed of feedforward neural networks is in general far slower than required and it has been a major bottleneck in their applications for past decades. Two key reasons behind may be: (1) the slow gradient-based learning algorithms are extensively used to train neural networks, and (2) all the parameters of the networks are tuned iteratively by using such learning algorithms. Unlike these conventional implementations, this paper proposes a new learning algorithm called **e**xtreme **l**earning **m**achine (ELM) for **s**ingle-hidden **l**ayer **f**eedforward neural **n**etworks (SLFNs) which randomly chooses hidden nodes and analytically determines the output weights of SLFNs. In theory, this algorithm tends to provide good generalization performance at extremely fast learning speed. The experimental results based on a few artificial and real benchmark function approximation and classification problems including very large complex applications show that the new algorithm can produce good generalization performance in most cases and can learn thousands of times faster than conventional popular learning algorithms for feedforward neural networks.[1]

Keywords: Feedforward neural networks; Back-propagation algorithm; Extreme learning machine; Support vector machine; Real-time learning; Random node

1. Introduction

Feedforward neural networks have been extensively used in many fields due to their ability: (1) to approximate complex nonlinear mappings directly from the input samples; and (2) to provide models for a large class of natural and artificial phenomena that are difficult to handle using classical parametric techniques. On the other hand, there lack faster learning algorithms for neural networks. The traditional learning algorithms are usually far slower than required. It is not surprising to see that it may take several hours, several days, and even more time to train neural networks by using traditional methods.

From a mathematical point of view, research on the approximation capabilities of feedforward neural networks has focused on two aspects: universal approximation on compact input sets and approximation in a finite set of training samples. Many researchers have explored the universal approximation capabilities of standard multilayer feedforward neural networks. Hornik [7] proved that if the activation function is continuous, bounded and nonconstant, then continuous mappings can be approximated in measure by neural networks over compact input sets. Leshno [17] improved the results of Hornik [7] and proved that feedforward networks with a nonpolynomial activation function can approximate (in measure) continuous functions. In real applications, the neural networks are trained in finite training set. For function approximation in a finite training set, Huang and Babri [11] shows that a **s**ingle-hidden **l**ayer **f**eedforward neural **n**etwork (**SLFN**) with at most N hidden nodes and with almost any nonlinear activation function can exactly learn N distinct observations. It should be noted that the input weights (linking the input layer to the first hidden layer) and hidden layer biases need to be adjusted in all these previous theoretical research works as well as in almost all practical learning algorithms of feedforward neural networks.

*Corresponding author. Tel.: +65 6790 4489; fax: +65 6793 3318.
E-mail address: egbhuang@ntu.edu.sg (G.-B. Huang).
URL: http://www.ntu.edu.sg/home/egbhuang/.

[1]For the preliminary idea of the ELM algorithm, refer to "Extreme Learning Machine: A New Learning Scheme of Feedforward Neural Networks", Proceedings of International Joint Conference on Neural Networks (IJCNN2004), Budapest, Hungary, 25–29 July, 2004.

0925-2312/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2005.12.126

Traditionally, all the parameters of the feedforward networks need to be tuned and thus there exists the dependency between different layers of parameters (weights and biases). For past decades, gradient descent-based methods have mainly been used in various learning algorithms of feedforward neural networks. However, it is clear that gradient descent-based learning methods are generally very slow due to improper learning steps or may easily converge to local minima. And many iterative learning steps may be required by such learning algorithms in order to obtain better learning performance.

It has been shown [23,10] that SLFNs (with N hidden nodes) with randomly chosen input weights and hidden layer biases (and such hidden nodes can thus be called random hidden nodes) can exactly learn N distinct observations. Unlike the popular thinking and most practical implementations that all the parameters of the feedforward networks need to be tuned, one may not necessarily adjust the input weights and first hidden layer biases in applications. In fact, some simulation results on artificial and real large applications in our work [16] have shown that this method not only makes learning extremely fast but also produces good generalization performance.

In this paper, we first rigorously prove that the input weights and hidden layer biases of SLFNs can be randomly assigned if the activation functions in the hidden layer are infinitely differentiable. After the input weights and the hidden layer biases are chosen randomly, SLFNs can be simply considered as a linear system and the output weights (linking the hidden layer to the output layer) of SLFNs can be *analytically determined* through simple generalized inverse operation of the hidden layer output matrices. Based on this concept, this paper proposes a simple learning algorithm for SLFNs called **e**xtreme **l**earning **m**achine (**ELM**) whose learning speed can be thousands of times faster than traditional feedforward network learning algorithms like back-propagation (BP) algorithm while obtaining better generalization performance. Different from traditional learning algorithms the proposed learning algorithm not only tends to reach the smallest training error but also the smallest norm of weights. Bartlett's [1] theory on the generalization performance of feedforward neural networks states for feedforward neural networks reaching smaller training error, the smaller the norm of weights is, the better generalization performance the networks tend to have. Therefore, the proposed learning algorithm tends to have good generalization performance for feedforward neural networks.

As the new proposed learning algorithm can be easily implemented, tends to reach the smallest training error, obtains the smallest norm of weights and the good generalization performance, and runs extremely fast, in order to differentiate it from the other popular SLFN learning algorithms, it is called the extreme learning machine in the context of this paper.

This paper is organized as follows. Section 2 rigorously proves that the input weights and hidden layer biases of SLFNs can be randomly assigned if the activation functions in the hidden layer are infinitely differentiable. Section 3 further proposes the new ELM learning algorithm for single-hidden layer feedforward neural networks (SLFNs). Performance evaluation is presented in Section 4. Discussions and conclusions are given in Section 5. The Moore–Penrose generalized inverse and the minimum norm least-squares solution of a general linear system which play an important role in developing our new ELM learning algorithm are briefed in the Appendix.

2. Single hidden layer feedforward networks (SLFNs) with random hidden nodes

For N arbitrary distinct samples $(\mathbf{x}_i, \mathbf{t}_i)$, where $\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{in}]^{\mathrm{T}} \in \mathbf{R}^n$ and $\mathbf{t}_i = [t_{i1}, t_{i2}, \ldots, t_{im}]^{\mathrm{T}} \in \mathbf{R}^m$, standard SLFNs with $\tilde{N}$ hidden nodes and activation function $g(x)$ are mathematically modeled as

$$\sum_{i=1}^{\tilde{N}} \beta_i g_i(\mathbf{x}_j) = \sum_{i=1}^{\tilde{N}} \beta_i g(\mathbf{w}_i \cdot \mathbf{x}_j + b_i) = \mathbf{o}_j, \quad j = 1, \ldots, N, \tag{1}$$

where $\mathbf{w}_i = [w_{i1}, w_{i2}, \ldots, w_{in}]^{\mathrm{T}}$ is the weight vector connecting the ith hidden node and the input nodes, $\beta_i = [\beta_{i1}, \beta_{i2}, \ldots, \beta_{im}]^{\mathrm{T}}$ is the weight vector connecting the ith hidden node and the output nodes, and b_i is the threshold of the ith hidden node. $\mathbf{w}_i \cdot \mathbf{x}_j$ denotes the inner product of $\mathbf{w}_i$ and $\mathbf{x}_j$. The output nodes are chosen linear in this paper.

That standard SLFNs with $\tilde{N}$ hidden nodes with activation function $g(x)$ can approximate these N samples with zero error means that $\sum_{j=1}^{\tilde{N}} \|\mathbf{o}_j - \mathbf{t}_j\| = 0$, i.e., there exist β_i, $\mathbf{w}_i$ and b_i such that

$$\sum_{i=1}^{\tilde{N}} \beta_i g(\mathbf{w}_i \cdot \mathbf{x}_j + b_i) = \mathbf{t}_j, \quad j = 1, \ldots, N. \tag{2}$$

The above N equations can be written compactly as

$$\mathbf{H}\beta = \mathbf{T}, \tag{3}$$

where

$$\mathbf{H}(\mathbf{w}_1, \ldots, \mathbf{w}_{\tilde{N}}, b_1, \ldots, b_{\tilde{N}}, \mathbf{x}_1, \ldots, \mathbf{x}_N) = \begin{bmatrix} g(\mathbf{w}_1 \cdot \mathbf{x}_1 + b_1) & \cdots & g(\mathbf{w}_{\tilde{N}} \cdot \mathbf{x}_1 + b_{\tilde{N}}) \\ \vdots & \cdots & \vdots \\ g(\mathbf{w}_1 \cdot \mathbf{x}_N + b_1) & \cdots & g(\mathbf{w}_{\tilde{N}} \cdot \mathbf{x}_N + b_{\tilde{N}}) \end{bmatrix}_{N \times \tilde{N}}, \tag{4}$$

$$\beta = \begin{bmatrix} \beta_1^{\mathrm{T}} \\ \vdots \\ \beta_{\tilde{N}}^{\mathrm{T}} \end{bmatrix}_{\tilde{N} \times m} \quad \text{and} \quad \mathbf{T} = \begin{bmatrix} \mathbf{t}_1^{\mathrm{T}} \\ \vdots \\ \mathbf{t}_N^{\mathrm{T}} \end{bmatrix}_{N \times m}. \tag{5}$$

As named in Huang et al. [11,10], $\mathbf{H}$ is called the hidden layer output matrix of the neural network; the ith column of $\mathbf{H}$ is the ith hidden node output with respect to inputs $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$.