Neurocomputing 74 (2011) 2491–2501

Contents lists available at ScienceDirect

Neurocomputing

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

A new robust training algorithm for a class of single-hidden layer feedforward neural networks

Zhihong Man [a,*], Kevin Lee [a], Dianhui Wang [b], Zhenwei Cao [a], Chunyan Miao [c]

[a] Faculty of Engineering and Industrial Sciences, Swinburne University of Technology, Vic. 3122, Australia
[b] Department of Computer Science and Computer Engineering, La Trobe University, Vic. 3086, Australia
[c] School of Computer Engineering, Nanyang Technological University, Singapore 639798, Singapore

ARTICLE INFO

Available online 12 May 2011

Keywords:
Linear FIR filter
Extreme learning machine
Feedforward neural networks
Signal processing

ABSTRACT

A robust training algorithm for a class of single-hidden layer feedforward neural networks (SLFNs) with linear nodes and an input tapped-delay-line memory is developed in this paper. It is seen that, in order to remove the effects of the input disturbances and reduce both the structural and empirical risks of the SLFN, the input weights of the SLFN are assigned such that the hidden layer of the SLFN performs as a pre-processor, and the output weights are then trained to minimize the weighted sum of the output error squares as well as the weighted sum of the output weight squares. The performance of an SLFN-based signal classifier trained with the proposed robust algorithm is studied in the simulation section to show the effectiveness and efficiency of the new scheme.

1. Introduction

The applications of single-hidden layer feedforward neural networks (SLFNs) have been receiving a great deal of attention in many engineering disciplines. As shown in [1–17], by properly choosing the number of nodes in both the hidden layer and the output layer and training the input and the output weights, one may use SLFNs for function approximation, digital signal and image processing, complex system modeling, adaptive control, data classification and information retrieval. In practical applications, the techniques for training the weights of SLFNs are very important in order to guarantee the good performance of SLFNs. The most popular training technique used for SLFNs is the gradient-based back-propagation (BP) algorithm [1,2]. It has been seen that the BP can be easily implemented from the output layer to the hidden layer of SLFNs in real-time. However, the slow convergence has limited the BP in many practical applications where the fast on-line training is required. In addition, the sensitivity of the SLFNs, trained using the BP, with respect to the input disturbances and the large spread of data is another important issue that needs to be further studied by the researchers and engineers in neural computing.

In [4–9], a learning algorithm called *extreme learning machine* (**ELM**) for SLFNs is proposed, where the input weights and the hidden layer biases of an SLFN are randomly assigned, the SLFN is then simply treated as a linear network and the output weights of the SLFN are then computed by using the generalized inverse of the hidden layer output matrix. It has been noted that the ELM has extremely fast learning speed and produces good performance in many cases. However, the poor robustness property of the SLFNs trained with the ELM has been observed as the SLFNs are used for signal processing to handle the noisy data. For instance, as the input weights and the hidden layer biases are randomly assigned in an SLFN, the changes of the hidden layer output matrix sometimes are very large because of the effects of the input disturbances, which also result in the big changes of the output weight matrix of the SLFN.

In [10,11], two modified ELM algorithms are proposed, where the cost function consists of the sum of the weighted error squares and the sum of the weighted output weight squares. In terms of the optimization of the cost function in the output weight space and the proper choice of the weights of the error squares, the structural and the empirical risks are balanced and reduced. However, the structural and the empirical risks are not significantly reduced and the robustness property of the trained SLFN is not significantly improved because of the random assignment of both the input weights and the hidden layer biases. According to the statistical learning theory [18–24], the big changes of the output weight matrix will largely increase both the structural risk and empirical risk of the SLFNs. Therefore, in order to substantially reduce the structural and empirical risks and improve the robustness property of SLFNs with respect to the input disturbances, the proper choice of the input weights of SLFNs is absolutely necessary.

* Corresponding author. Tel.: +61 3 9214 5175; fax: +61 3 9214 8264.
E-mail address: zman@swin.edu.au (Z. Man).

0925-2312/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2010.11.033

In this paper, we propose a new robust training algorithm for a class of SLFNs, with both linear nodes and an input tapped-delay-line memory, for signal processing purpose. Since the output of each linear hidden node in the SLFN is the sum of the weighted input data, each node can be treated as a FIR filter. Therefore, the hidden layer with linear nodes can be designed as the pre-processor of the input data. For instance, based on the FIR filter design techniques in signal processing [25–28], we may design the hidden layer as a group of low-pass filters or high-pass filters or band-pass filters or band-stop filters or other types of filters for the purpose of the pre-processing of the input data with disturbances and undesired frequency components. The advantages of the hidden layer's pre-processing function are that not only the input disturbances and the undesired frequency components can be removed, but also both the structural and empirical risks of the SLFNs can be greatly reduced as well from the viewpoint of the output of the SLFNs.

For the design of the output weight matrix of the SLFNs, in this paper, we choose an objective function which includes both the weighted sum of the output error squares and the weighted sum of the output weight squares of the SLFNs [1,10,11,29–31]. By minimizing this objective function in the output weight space, as well as the proper choice of the input weights based on the FIR filter design techniques, both the structural and empirical risks can be balanced and reduced for signal processing purpose. For the comparison with the ELM and the modified ELM algorithms in [10,11], we call the new training scheme to be developed in this paper as the **FIR-ELM** algorithm. According to the ELM theory, the hidden nodes used in SLFNs may not be neuron alike. We may alternatively call the hidden linear nodes, with the input weights trained with the FIR filtering techniques in this paper, as the **FIR nodes**, which are one type of the many possible hidden nodes mentioned in [4–10].

It should be emphasized that the SLFNs considered in [4–11] use the nonlinear hidden nodes and the linear output nodes without dynamics and, by the proper choice of the output weights, the SLFNs can uniformly approximate nonlinear input–output mappings. However, the class of SLFNs considered in this paper use both the linear hidden nodes and the linear output nodes. In order to make such SLFNs to have the universal approximation capability, an input tapped-delay-line memory is added to the input layer. It has been shown in [32] that the SLFN with linear (or nonlinear) nodes, as well as an input tapped-delay-line memory, is capable of approximating the maps that are causal, time invariant and satisfy certain continuity and approximately-finite-memory conditions. Since, in many cases of signal processing, the input and output data have some dynamic relationships, it is thus convenient to train the SLFNs with linear nodes as well as an input tapped-delay-line memory to perform as the signal processors.

The rest of the paper is organized as follows: In Section 2, a class of SLFNs, with linear nodes and an input tapped-delay-line memory, as the signal classifiers are formulated, and the issues on the empirical and the structural risks, as well as the robustness property of the SLFNs with respect to the input disturbances, trained with the ELM algorithm and the modified ELM algorithm in [4–11], are studied. In Section 3, the design of the input weights using FIR filtering technique, for reducing both the empirical and structural risks, improving the robustness of the SLFNs with respect to the input disturbances is presented and removing some undesired frequency components. In Section 4, the design of the output weights by the minimization of the weighted sum of the output error squares as well as the weighted sum of the output weight squares of the SLFNs are discussed in detail. In Section 5, the SLFN-based signal classifiers, trained with the ELM in [4–9], the modified ELM in [11] and the FIR-ELM developed in this paper are simulated and compared in support of the effectiveness of the proposed FIR-ELM algorithm for signal processing. Section 6 gives conclusions and some further work.

2. Problem formulation

The architecture of a class of SLFNs with the linear hidden nodes and an input tapped-delay-line memory is presented in Fig. 1, where the output layer has m linear nodes, the hidden layer has $\tilde{N}$ linear nodes, D is the unit-delay element, the $n-1$ time-delay elements, added to the input of the neural network, form the tagged-delay-line memory, which indicates that the input sequence $x(k)$, $x(k-1)$, …, $x(k-n+1)$ represent a time series consisting of the present observation $x(k)$ and the $n-1$ past observations of the process.

From Fig. 1, the input data vector $\mathbf{x}(k)$ and the output data vector $\mathbf{O}(k)$ can be expressed as follows:

$$\mathbf{x}(k) = \begin{bmatrix} x(k) & x(k-1) & \cdots & x(k-n+1) \end{bmatrix}^T \tag{2.1}$$

$$\mathbf{O}(k) = \begin{bmatrix} o_1(k) & o_2(k) & \cdots & o_m(k) \end{bmatrix}^T \tag{2.2}$$

the output of the ith hidden neuron is computed as

$$y_i = \sum_{j=1}^{n} w_{ij} x(k-j+1) = w_i^T x(k) \quad \text{for} \quad i = 1,\ldots,\tilde{N} \tag{2.3}$$

with

$$\mathbf{w}_i = \begin{bmatrix} w_{i1} & w_{i2} & \cdots & w_{in} \end{bmatrix}^T \quad \text{for} \quad i = 1,\ldots,\tilde{N} \tag{2.4}$$

and the ith output of the neural network, $o_i(k)$, is of the form

$$o_i(k) = \sum_{p=1}^{\tilde{N}} \beta_{pi} \mathbf{w}_p^T \mathbf{x}(k) \quad \text{for} \quad i = 1,\ldots,m \tag{2.5}$$

Thus, the output data vector $\mathbf{O}(k)$ can be expressed as

$$\mathbf{O}(k) = \sum_{p=1}^{\tilde{N}} \boldsymbol{\beta}_p \mathbf{w}_p^T \mathbf{x}(k) \tag{2.6}$$

with

$$\boldsymbol{\beta}_i = \begin{bmatrix} \beta_{i1} & \beta_{i2} & \cdots & \beta_{im} \end{bmatrix}^T \quad \text{for} \quad i = 1,\ldots,\tilde{N} \tag{2.7}$$

In this paper, we use N distinct sample signal data vector pairs $(\mathbf{x}_i, \mathbf{t}_i)$ to train the SLFN given in Fig. 1, where $\mathbf{x}_i =$

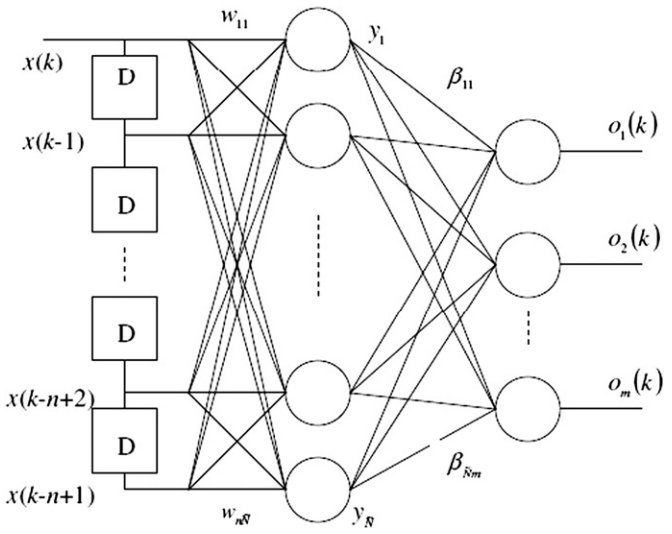

Fig. 1. A single-hidden layer neural network with linear nodes.