Neural Networks 22 (2009) 1471–1483

Contents lists available at ScienceDirect

Neural Networks

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

A novel nonlinear adaptive filter using a pipelined second-order Volterra recurrent neural network

Haiquan Zhao *, Jiashu Zhang

Si-Chuan Province, Key Lab of Signal and Information Processing, Southwest Jiaotong University, Chengdu, 610031, China

ARTICLE INFO

Article history:
Received 14 December 2008
Received in revised form 19 March 2009
Accepted 18 May 2009

Keywords:
Recursive second-order Volterra filter
Recurrent neural network
Pipelined recurrent neural network
Pipelined architecture
Adaptive Volterra filter

ABSTRACT

To enhance the performance and overcome the heavy computational complexity of recurrent neural networks (RNN), a novel nonlinear adaptive filter based on a pipelined second-order Volterra recurrent neural network (PSOVRNN) is proposed in this paper. A modified real-time recurrent learning (RTRL) algorithm of the proposed filter is derived in much more detail. The PSOVRNN comprises of a number of simple small-scale second-order Volterra recurrent neural network (SOVRNN) modules. In contrast to the standard RNN, these modules of a PSOVRNN can be performed simultaneously in a pipelined parallelism fashion, which can lead to a significant improvement in its total computational efficiency. Moreover, since each module of the PSOVRNN is a SOVRNN in which nonlinearity is introduced by the recursive second-order Volterra (RSOV) expansion, its performance can be further improved. Computer simulations have demonstrated that the PSOVRNN performs better than the pipelined recurrent neural network (PRNN) and RNN for nonlinear colored signals prediction and nonlinear channel equalization. However, the superiority of the PSOVRNN over the PRNN is at the cost of increasing computational complexity due to the introduced nonlinear expansion of each module.

1. Introduction

Since many signals (such as nonlinear colored signals, speech signals, distortion in nonlinear channels in communication systems, *etc.*) are generated from an inherently nonlinear physical mechanism and have statistically nonstationary properties, linear adaptive techniques-based the linear model do not perform well. Therefore, the nonlinearities must be accounted for in the design of adaptive filters. Because the prime advantages of neural networks are: their ability to learn based on optimization of an appropriate error function and their excellent performance for the approximation of nonlinear functions, different types of adaptive nonlinear filters-based neural networks have been proposed and applied in many papers in the literature (Mandic, 2001; Narendra & Parthasarathy, 1990).

Among adaptive nonlinear filter-based neural networks, recurrent neural networks (RNNs), widely applied in nonlinear signal processing fields, have shown better performance than feedforward neural networks (Mandic, 2001). The RNN is a dynamic network due to its feedback in nature, while feedforward neural networks with multilayer architecture represent static nonlinear models (Narendra & Parthasarathy, 1990). Moreover, RNNs can yield smaller structures than nonrecursive neural networks in the same way that infinite impulse response (IIR) filters can replace longer finite impulse response (FIR) filters. Therefore, the local/global recurrence and internal/external feedback of the RNNs enable them to acquire accurately nonlinear models, which make it suitable for nonlinear prediction, modeling and channel equalization (Choi, Antonio, Lima, & Haykin, 2005; Choi, Bouchard, & Yeap, 2005; Connor, Martin, & Atlas, 1994; Hacioglu, 1997; Han, Xi, Xu, & Yin, 2004; Kechriotis & Manolakos, 1994; Kechriotis, Zervas, & Manolakos, 1994; Mandic, 2001; Parisi, Claudio, Orlandi, & Rao, 1997; Williams & Zipser, 1989). In 1989, Williams and Zipser firstly proposed a fully connected RNN trained by a real-time recurrent learning (RTRL) algorithm (Williams & Zipser, 1989). Following their work, a robust learning algorithm proposed is applied to a RNN for approximating the NARMA processes in 1994. Although these algorithms-based gradient descents exhibit the lack of the instability, RNNs still show more powerful performance than feedforward neural networks (Connor et al., 1994). Kechriotis *etc.* have successfully applied the RNN to solve the nonlinear channel equalization problem (Kechriotis et al., 1994). Kechriotis and Manolakos (1994) presented the fully RNN with complex weights. Hacioglu (1997) introduced a method of extending RNN equalizers to M-PAM signal reconstruction in the presence of ISI and AWGN. The literature (Parisi et al., 1997) describes a novel approach to learning in a RNN that exploits the principle of discriminative learning, minimizing an error functional that is a direct measure of the classification error. Its main feature is a higher speed of convergence. In 2004, a new methodology to model and predict chaotic time series based on a new recurrent predictor neural network is studied

* Corresponding author.
E-mail address: hqzhao0815@yahoo.com.cn (H. Zhao).

0893-6080/$ – see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2009.05.010

in Han et al. (2004). Recently, an extended Kalman filter (EKF) and unscented Kalman filter (UKF) algorithms for the RNN equalizer introduced in Choi et al. (2005) and Choi et al. (2005) have been successfully applied in time-variant and time-invariant nonlinear channel equalization. Although many types of adaptive nonlinear filter based on the RNN using the various algorithms can achieve a fast convergence speed, good tracking performance and highly filtered accuracy, the heavy computational complexity can severely limit the RNN in implementation.

To reduce the heavy computational burdens of a RNN, a novel computationally efficient modular nonlinear filter using PRNN was presented by Haykin and Li (1995). The design of such a modular network is based on the principle of divide and conquers, that is, a complex RNN with a large number of neurons can be divided into a number of simpler small-scale RNN models (Haykin & Li, 1995; Li & Haykin, 1993). Since those modules of PRNN can be performed simultaneously in a pipelined parallelism fashion, this results in a significant improvement in its total computational efficiency. Moreover, due to the modular nesting, the performance can be improved to a certain extent. Therefore, the PRNN has been successfully used for a variety of applications where complexity and nonlinearity pose major problems, including speech processing (Haykin & Li, 1995), ATM traffic modeling (Chang & Hu, 1997), and communications (Chang & Hu, 1999; Chen, Chang, & Hsieh, 2006). In 1998, an extended recursive least squares (ERLS) learning algorithm of PRNN was introduced to improve the performance of adaptive speech prediction (Baltersee & Chambers, 1998). Recently, for the optimal selection of the parameters of PRNN, the authors elaborate on the cost function used for learning, suggesting different forms of module weighting factor (Chen, Gautama, & Mandic, 2008; Mandic & Chambers, 2000, 1999). For complex-valued nonlinear and nonstationary signals, a complex-valued nonlinear adaptive filter using a pipelined recurrent neural network (CPRNN) is presented (Goh & Mandic, 2005). Considering the pipelined architecture and the learning capabilities of a recurrent fuzzy neural network (RFNN), a class of pipelined recurrent fuzzy neural networks (PRFNN) proposed for nonlinear adaptive speech prediction can provide considerably better performance compared to PRNN (Stavrakoudis & Theocharis, 2007). Nevertheless, PRNN and RNN are both confronted with the same problems: they utilize a linear input and first-order recurrent term only while they fail to utilize the high-order terms of inputs. Hence, the performance of the PRNN and the RNN is limited by a nonlinear processing capability and should be further enhanced.

It is well known that the polynomial filter may be interpreted as an extension version of linear filter to the nonlinear case. Moreover, the key attractive feature is that the truncated Volterra filter can deal with a general class of nonlinear systems, while its output is still linear with respect to various higher order kernels or impulse responses. The major drawback of using the Volterra filter is that its computational complexity is much higher than the linear adaptive filter since the adaptive Volterra filter requires a large number of multidimensional coefficients to accurately model nonlinear systems (Mathews, 1991). Moreover, the computational complexity increases exponentially as the order of the polynomials. To overcome the limitation of the computational complexity, only the second order Volterra (SOV) filter and third-order Volterra (TOV) filter can be used for implementation in practice. However, the adaptive SOV filter cannot accurately model the systems that have strong nonlinearity (such as strongly saturated signals) with a reasonable filter length, and the TOV filter can improve the nonlinear processing capability to a certain extent at the cost of more computational complexity. The adaptive recursive second order Volterra (RSOV) filter, which is a nonlinear extension of the IIR filter, can be seen as an alternative solution to higher-order Volterra filters when the nonlinearities produced by the system are directly dependent on the lower ones, in keeping a second-order computational burden. Moreover, the adaptive RSOV filter can model nonlinear systems accurately with a lower order than the higher Volterra filters that use only the feedforward coefficients (Roy, Stewart, & Durrani, 1996a, 1996b).

As a consequence, by combining the pipelined architecture type of the PRNN and the characteristics of RSOV, a novel adaptive nonlinear filter using the PSOVRNN is presented to improve the performance and overcome the heavy computational complexity of the RNN in this paper. Since each module of the PSOVRNN is a SOVRNN in which nonlinearity is introduced by enhanced the input pattern with the RSOV expansion (Roy et al., 1996a, 1996b), the nonlinear processing capability of the PSOVRNN is enhanced. At the same time, in contrast to the RNN, the computational complexity is further reduced by using pipelined architecture.

This paper is organized as follows. The RNN is introduced in Section 2. Section 3 presents the proposed novel adaptive nonlinear filter. The adaptive algorithms of the PSOVRNN are deduced in Section 4. In Section 5, the convergence performance and stability conditions are discussed. The computational complexity of the PSOVRNN is analyzed in Section 6. Section 7 provides the effectiveness of the proposed nonlinear filter illustrated by comparing with the PRNN and RNN filters. Section 8 is devoted to a brief summary and discussion.

2. The recurrent neural network

A fully connected recurrent neural network, consisting of q neurons with p external inputs and q feedback connections, is depicted in Fig. 1. Let $y_l(n)$ denote the output of a neuron $l = 1, \dots, q$ at time index n and $s(n) = [x(n-1), \dots, x(n-p)]^T$ the $(1 \times p)$ external input vector. Then the overall input $P(n)$ to the network represents a concatenation of vectors $s(n)$, the bias input 1 and $y(n) = [y_1(n-1), \dots, y_q(n-1)]^T$ is given by

$$\begin{aligned} P(n) &= [P_1(n), \dots, P_{p+q+1}(n)]^T \\ &= [s^T(n), 1, y^T(n)]^T \\ &= [x(n-1), \dots, x(n-p), 1, y_1(n-1), \dots, y_q(n-1)]^T \end{aligned} \quad (1)$$

where $(\cdot)^T$ denotes the vector transpose operator.

For the lth neuron, its weights form a $(p+q+1)\times 1$ dimensional weight vector $W_l(n) = [w_{1,l}(n), \dots, w_{p+q+1,l}(n)]^T$, $l = 1, \dots, q$. Which are encompassed in the weight matrix of the network $W(n) = [W_1(n), \dots, W_q(n)]^T$.

The output $y_l(n)$ of every neuron can be expressed as

$$y_l(n) = \phi(net_l(n)), \quad l = 1, \dots, q \quad (2)$$

where ϕ is a nonlinear activation function of a neuron and

$$net_l(n) = \sum_{k=1}^{p+q+1} w_{k,l}(n) P_k(n) \quad (3)$$

is the net input to lth neuron node at time index n. For simplicity, we can rewrite the above equation as follows:

$$net_l(n) = W^T(n)P(n). \quad (4)$$

Thus, $y_1(n)$is as the final output of the entire RNN.