Multiresolution FIR Neural-Network-Based Learning Algorithm Applied to Network Traffic Prediction

Vicente Alarcon-Aquino, *Member, IEEE,* and Javier A. Barria, *Member, IEEE*

***Abstract*—In this paper, a multiresolution finite-impulse-response (FIR) neural-network-based learning algorithm using the maximal overlap discrete wavelet transform (MODWT) is proposed. The multiresolution learning algorithm employs the analysis framework of wavelet theory, which decomposes a signal into wavelet coefficients and scaling coefficients. The translation-invariant property of the MODWT allows aligment of events in a multiresolution analysis with respect to the original time series and, therefore, preserving the integrity of some transient events. A learning algorithm is also derived for adapting the gain of the activation functions at each level of resolution. The proposed multiresolution FIR neural-network-based learning algorithm is applied to network traffic prediction (real-world aggregate Ethernet traffic data) with comparable results. These results indicate that the generalization ability of the FIR neural network is improved by the proposed multiresolution learning algorithm.**

***Index Terms*—Finite-impulse-response (FIR) neural networks, multiresolution learning, network traffic prediction, wavelet transforms, wavelets.**

I. Introduction

WHEN designing adaptive congestion control and proactive management schemes for communication networks, predicting the behavior of the network traffic is very important. Predicting normally relies on the construction of stochastic models to predict subsequent time series values given a history of past values. This has traditionally been performed by the use of linear models and neural networks. These models are known as global models or single-resolution learning techiques since only one model is used to characterize the measured process [2], [16]. The classical linear models used for time-series prediction include the auto-regressive (AR) and the auto-regressive moving average (ARMA) models (see, e.g., [4]). These models are applied in most of the reported cases to stationary time series, that is, series whose statistical properties do not change with time [9]. The parameters of these models can be estimated in blocks or in a sequence manner with the least-mean square (LMS) and the recursive-least square (RLS) algorithms [8]. These linear procedures are based on *a priori* information about the statistics of the data to be processed. Artificial neural networks (ANNs), on the other hand, do not require *a priori* information about the statistics and have demonstrated a great potential for time-series prediction (see, e.g., [9], [10], [13], [25], and [26]). Moreover, neural networks can naturally account for the practical realities of nonlinearity, nonstationarity, and non-Gaussianity normally found in real data [9].

Feedforward neural networks, also known as multilayer perceptron (MLP) networks, have been recognized for their approximation capability of unknown functions [10]. MLPs, however, are static networks which simply map input to output, and are incapable of processing temporal information [10], [25], [26]. The simplest method for predicting time series in MLP networks is to provide its time-delayed samples to the input layer (see, e.g., [9], [13], and [26]). Note, however, that time-delayed samples can be applied at the network inputs only; that is, the dynamics are external to the actual network itself. Finite-impulse-response (FIR) neural networks represent a generalization of MLP networks in which scalar static weights are replaced by adaptive FIR linear filters [25]. Note that FIR neural networks achieved the best performance in the task of time-series prediction when compared with standard recurrent neural networks, linear predictors, Wiener filters, and feedforward neural networks (see, e.g., [25] and [26]). Although recurrent neural networks have been suggested recently as viable prediction tools [19] in the work reported here, we use the well-established feedforward neural network tool. The aim is to prove the underlying fundamentals of our approach. It is worth noting that the best neural-network architecture is problem dependent and that the efficiency of the learning algorithm is a more important factor than the network model used [7], [11].

Case-based reasoning systems and neural-network techniques are used frequently in the forecasting literature [27], [29], [32], [35] and have found broad applications in telecommunications such as congestion control in ATM [30] and fraud detection [31]. Recently, it has been reported that multiresolution learning can significantly improve neural network's generalization performance (generalization refers to the ability of the neural network to provide a satisfactory performance in response to test data never seen by the network before [10]) and neural-network robustness on difficult signal prediction tasks (see, e.g., [16] and [17]). Note that the approach reported in [16] and [17] is still based on the decimated discrete wavelet transform which can only be applied to time series with a sample size N equal to 2^J, where J denotes the number of resolutions. Furthermore, it can introduce ambiguities in the time domain due to the decimation process that needs to be applied at the output of the corresponding filters (see, e.g., [21]). Similar multiresolution learning approaches have been

Manuscript received October 23, 2003; revised September 3, 2004. This work was supported in part by the National Council for Science and Technology (CONACYT), Mexico. This paper was recommended by Associate Editor J. Wang.

V. Alarcon-Aquino is with the Department of Electrical and Electronic Engineering, Universidad de las Americas-Puebla, Puebla CP 72820, Mexico (e-mail: vicente.alarcon@udlap.mx).

J. A. Barria is with the Department of Electrical and Electronic Engineering, Imperial College London, London SW7 2BT, U.K. (e-mail: j.barria@imperial.ac.uk).

Digital Object Identifier 10.1109/TSMCC.2004.843217

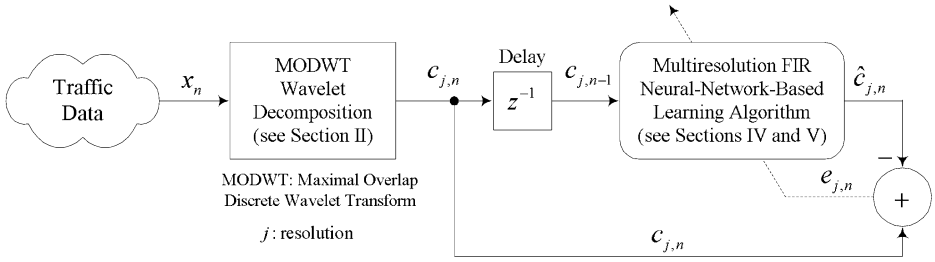

Fig. 1. Multiresolution learning algorithm for the FIR neural-network predictor.

reported in [23] and [28]. Support vector machines experts using a self-organizing feature map (SOM) have also been considered for time-series prediction (see, e.g., [5] and [36]). Note that the algorithms used in [16], [17], [23], and [28] are based on static MLP networks using the standard backpropagation algorithm in contrast with our proposed algorithm that uses FIR neural networks using temporal backpropagation [25]. As stated in [16], multiresolution learning can reveal the correlation structure (low- and high-frequency components) at each level of resolution that may be obscured in the original signal, and employs the analysis framework of wavelet theory [6], [18], which decomposes a signal into wavelet coefficients and scaling coefficients (see Section II).

This paper presents a multiresolution FIR neural-network-based learning algorithm using the maximal overlap discrete wavelet transform (MODWT) with application to network traffic prediction. The contributions of this paper can be summarized as follows. First, a multiresolution learning algorithm is proposed for time-series prediction based on FIR neural networks [25] and the MODWT [21]. The MODWT can be applied to any sample size, and the wavelet coefficients are translation-invariant. This property allows alignment of events in a multiresolution analysis with respect to the original time series and, therefore, preserving the integrity of some transient events. Note that the proposed algorithm has its foundations on the multiresolution learning paradigm reported in [16]. Second, to increase the learning speed and produce a good generalization performance on unseen data, a learning algorithm is derived for adapting the gain of the activation functions at each level of resolution based on the temporal backpropagation algorithm [25]. Finally, the proposed multiresolution FIR neural-network-based learning algorithm is applied to network traffic prediction (real-world aggregate Ethernet traffic data), which is known to be complex and irregular, suggesting that it presents a difficult signal prediction problem [16], [22].

The basic idea of the multiresolution FIR neural-network (MFIRNN)-based learning algorithm is as follows. First, the signal x_n is decomposed into wavelet coefficients and scaling coefficients using the MODWT (Fig. 1). The input of the MFIRNN-based learning algorithm is the known value of $c_{j,n-1}$ (scaling coefficients) and the output $\hat{c}_{j,n} = \mathcal{H}_q(c_{j,n-1})$ is the single step estimate of the true time-series value $c_{j,n}$, where $\mathcal{H}_q$ is the MFIRNN with total memory length q, and j represents the level of resolution. The model for learning can thus be represented by

$$c_{j,n} = \mathcal{H}(c_{j,n-1}, c_{j,n-2}, \ldots, c_{j,n-q}) + e_{j,n} \tag{1}$$

where $j = 1, 2, \ldots, J$ denotes the levels of resolution. During training, the objective is therefore to minimize the squared error $e_{j,n}^2 = (c_{j,n} - \hat{c}_{j,n})^2$ by using the multiresolution FIR neural-network-based learning algorithm (for details, see Section IV).

The remainder of this paper is organized as follows. In Section II, a review of the MODWT is presented. FIR neural networks are described in Section III. In Section IV, a multiresolution FIR neural network with adaptive gain parameter at each level of resolution is proposed. The multiresolution FIR neural-network-based learning algorithm is also derived. In Section V, experimental results and comparisons with previously published approaches are presented. In Section VI, conclusions of this paper are reported.

II. Maximal Overlap DWT

In this section, we introduce the MODWT to be used in the following sections. The computation of the DWT is based on discrete compactly supported filters of the Daubechies class [6]. The even-length scaling filter and the wavelet filter are denoted by $\{g_l : l = 0, 1, \ldots, L-1\}$ and $\{h_l : l = 0, 1, \ldots, L-1\}$, respectively, where L is the length of the filter. To build the MODWT, a rescaling of the defining filters is required to conserve energy, that is, $\tilde{g}_l = g_l/\sqrt{2}$ and $\tilde{h}_l = h_l/\sqrt{2}$, so that $\sum_{l=0}^{L-1} \tilde{g}_l^2 = 1/2$ and, therefore, the filters are still quadrature mirror filters (QMFs). The wavelet filter must satisfy the following properties (see, e.g., [21]):

$$\sum_{l=0}^{L-1} \tilde{h}_l = 0 \tag{2}$$

$$\sum_{l=0}^{L-1} \tilde{h}_l^2 = \frac{1}{2} \quad \text{and} \quad \sum_{l=-\infty}^{\infty} \tilde{h}_l \tilde{h}_{l+2r} = 0 \tag{3}$$

for all nonzero integers r. The scaling filter $\{\tilde{g}_l\}$ is also required to satisfy (3) and $\sum_{l=0}^{L-1} \tilde{g}_l = 1$. Now let $c_{j-1,n} = x_n$ be the time series, the MODWT pyramid algorithm generates the wavelet coefficients $\{d_{j,n}\}$ and the scaling coefficients $\{c_{j,n}\}$ from $\{c_{j-1,n}\}$ (Fig. 2). That is, with nonzero coefficients divided by $\sqrt{2}$ ([21]), the convolutions can be written as follows:

$$d_{j,n} = \sum_{l=0}^{L-1} \tilde{h}_l c_{j-1,(n-2^{j-1}l) \bmod N}$$

$$c_{j,n} = \sum_{l=0}^{L-1} \tilde{g}_l c_{j-1,(n-2^{j-1}l) \bmod N} \tag{4}$$

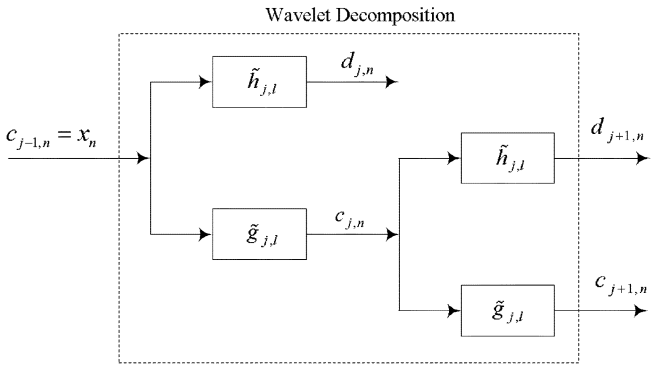

Fig. 2. Wavelet coefficients $\{d_{n,j}\}$ and scaling coefficients $\{c_{j,n}\}$ are computed by cascading convolutions with filters $\{\bar{h}_{j,l}, \bar{g}_{j,l}\}$.

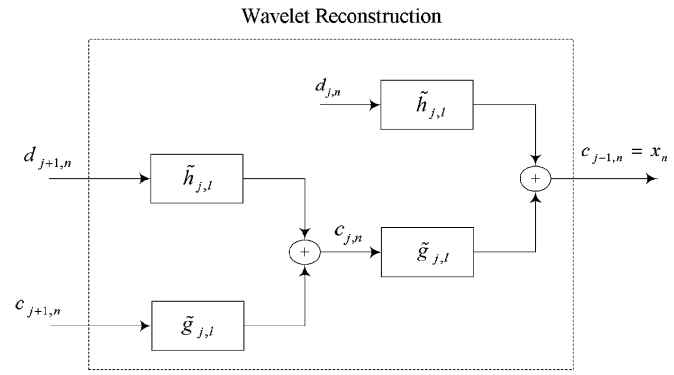

Fig. 3. Recovering the original signal. The filters $\{\bar{h}_{j,l}\}$ and $\{\bar{g}_{j,l}\}$ are used to compute the inverse MODWT.

where $n = 0, 1, \ldots, N-1$. Note that there is some difference in information about $c_{j-1,n} = x_n$ between the approximation at the level of resolution $j-1$ and the approximation at level of resolution j. This difference is the time-series detail needed to reconstruct the approximation at resolution $j-1$ from the approximation at resolution j (Fig. 3). Equation (4) can also be formulated as filter operations of the original time series $\{x_n\}$ using the filters $\{\tilde{h}_{j,l} = h_{j,l}/2^{j/2}\}$ and $\{\tilde{g}_{j,l} = h_{j,l}/2^{j/2}\}$, namely

$$d_{j,n} = \sum_{l=0}^{L_j-1} \tilde{h}_{j,l} x_{n-l \bmod N}$$
$$c_{j,n} = \sum_{l=0}^{L_j-1} \tilde{g}_{j,l} x_{n-l \bmod N} \tag{5}$$

where L_j is the length of the filter at level of resolution j. The MODWT wavelet coefficients $\{d_{j,n}\}$ at the level of resolution j are associated with the same nominal frequency band given by $|f| \in [1/2^{j+1}, 1/2^j]$. The original signal can be recovered from $\mathbf{d}_j$ and $\mathbf{c}_j$ using the inverse pyramid algorithm [21]

$$x_n = c_{j-1,n} = \sum_{l=0}^{L-1} \tilde{h}_l d_{j,n+2^{j-1} \bmod N} + \sum_{l=0}^{L-1} \tilde{g}_l c_{j,n+2^{j-1} \bmod N} \tag{6}$$

where $n = 0, 1, \ldots, N-1$ (Fig. 3). The MODWT scaling and wavelet coefficients computed by (4)–(6) are used in the following sections.

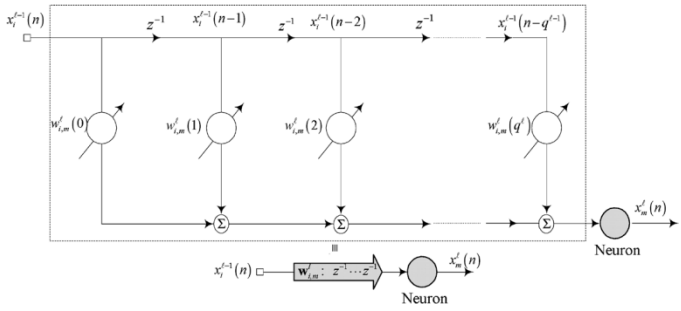

Fig. 4. TDL filter structure for an FIR linear filter (z^{-1} denotes a unit delay operator) with neuron.

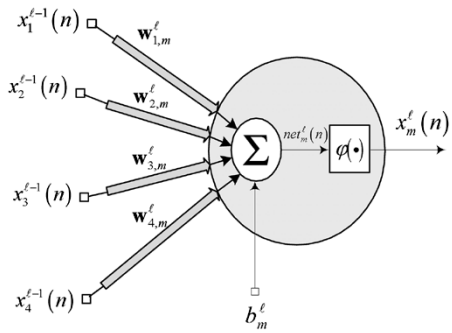

Fig. 5. FIR neuron model.

III. Review of FIR Neural Networks

MLP neural networks commonly utilize scalar static weights. MLP neural networks were originally proposed as a nonlinear autoregressive model for time-series prediction [13]. MLP networks, however, are incapable of processing temporal information [10], [25]. FIR neural networks, on the other hand, represent a generalization of MLP networks in which scalar static weights are replaced either by adaptive FIR linear filters or by infinite-impulse-response (IIR) linear filters (see, e.g., [1], [20], and [25]). The most basic filter that is modeled by a tapped-delay-line (TDL) as shown in Fig. 4 is used by the FIR network model.

The FIR neural model is shown in Fig. 5. In a FIR neural network, the coefficients of the FIR filter connecting neuron i to neuron m in layer ℓ are given by the vector

$$\mathbf{w}_{i,m}^{\ell} = \left[w_{i,m}^{\ell}(0), w_{i,m}^{\ell}(1), \ldots, w_{i,m}^{\ell}(q^{\ell})\right] \tag{7}$$

where q^{ℓ} is the number of lags in layer ℓ, and the vector of delayed activation values along with the FIR filter is given by

$$\mathbf{x}_i^{\ell-1}(n) = \left[x_i^{\ell-1}(n), x_i^{\ell-1}(n-1), \ldots, x_i^{\ell-1}(n-q^{\ell-1})\right]. \tag{8}$$

Note that each neuron passes the weighted sum of its inputs through an activation function $\varphi(\cdot)$, that is

$$\mathrm{net}_m^{\ell}(n) = \sum_i \mathbf{w}_{i,m}^{\ell} \cdot \mathbf{x}_i^{\ell-1}(n) + b_m^{\ell}$$
$$x_m^{\ell}(n) = \varphi(\mathrm{net}_m^{\ell}(n)) \tag{9}$$

where n denotes the discrete time index, the bias is denoted by b_m^{ℓ}, and the vector dot product $\mathbf{w}_{i,m}^{\ell} \cdot \mathbf{x}_i^{\ell-1}(n)$ denotes a filter operation.

To train the FIR neural network, we may unfold it in time. That is, the idea is to remove all time delays by expanding the network into a larger static equivalent structure, to which the standard backpropagation algorithm may be applied in the usual way. However, a more efficient and practical approach is to use the temporal backpropagation algorithm [25].

A. Temporal Back-Propagation Algorithm

Given an input sequence $\{\mathbf{x}(n)\}$, the network $\{\mathcal{H}\}$ produces the output sequence $\mathbf{y}(n) = \mathcal{H}\{W, \mathbf{x}(n)\}$, where $W = \{\mathbf{w}_{i,m}^{\ell}, \forall i, m, \ell\}$ represents the set of all filter coefficients in the network. The difference between the desired output $\mathbf{d}(n)$ at time n and the actual output of the network $\mathbf{y}(n)$ is the error $\{\mathbf{e}(n) = \mathbf{d}(n) - \mathbf{y}(n)\}$. The objective is to minimize the cost function $\xi = \sum_n \mathbf{e}^T(n)\mathbf{e}(n)$ with respect to W, where the sum is taken over all $n = 0, 1, \ldots, N_T$ points in the training sequence. The true gradient of the cost function is given as [25]

$$\frac{\partial \xi}{\partial \mathbf{w}_{i,m}^{\ell}} = \sum_{n=1}^{N_T} \frac{\partial \mathbf{e}^T(n)\mathbf{e}(n)}{\partial \mathbf{w}_{i,m}^{\ell}}. \tag{10}$$

An alternative expression for the true gradient is given by [25]

$$\frac{\partial \xi}{\partial \mathbf{w}_{i,m}^{\ell}} = \sum_{n=1}^{N_T} \frac{\partial \xi}{\partial \mathrm{net}_m^{\ell}(n)} \cdot \frac{\partial \mathrm{net}_m^{\ell}(n)}{\partial \mathbf{w}_{i,m}^{\ell}}. \tag{11}$$

This expansion results by considering the total cost ξ to be a function of $\mathrm{net}_m^{\ell}(n)$ over all indices of n. Then it follows that:

$$\frac{\partial \mathbf{e}^T(n)\mathbf{e}(n)}{\partial \mathbf{w}_{i,m}^{\ell}} \neq \frac{\partial \xi}{\partial \mathrm{net}_m^{\ell}(n)} \cdot \frac{\partial \mathrm{net}_m^{\ell}(n)}{\partial \mathbf{w}_{i,m}^{\ell}}. \tag{12}$$

That is, only the sums over all n are equivalent. By using the gradient descent method, the weights of the FIR filters are updated at each increment of time n according to

$$\mathbf{w}_{i,m}^{\ell}(n+1) = \mathbf{w}_{i,m}^{\ell}(n) - \eta_w \frac{\partial \xi}{\partial \mathrm{net}_m^{\ell}(n)} \cdot \frac{\partial \mathrm{net}_m^{\ell}(n)}{\partial \mathbf{w}_{i,m}^{\ell}} \tag{13}$$

where η_w denotes the rate of learning, and since $\mathrm{net}_m^{\ell}(n) = \sum_i \mathbf{w}_{i,m}^{\ell} \cdot \mathbf{x}_i^{\ell-1}(n)$, then it follows that:

$$\frac{\partial \mathrm{net}_m^{\ell}(n)}{\partial \mathbf{w}_{i,m}^{\ell}} = \frac{\partial (\mathbf{w}_{i,m}^{\ell} \cdot \mathbf{x}_i^{\ell-1}(n))}{\partial \mathbf{w}_{i,m}^{\ell}} = \mathbf{x}_i^{\ell-1}(n) \tag{14}$$

for all layers in the network. Defining $\partial \xi / \partial \mathrm{net}_m^{\ell}(n) = \delta_m^{\ell}(n)$, the final temporal backpropagation algorithm is then given by [25]

$$\mathbf{w}_{i,m}^{\ell}(n+1) = \mathbf{w}_{i,m}^{\ell}(n) - \eta_w \delta_m^{\ell}(n) \cdot \mathbf{x}_i^{\ell-1}(n) \tag{15}$$

with

$$\delta_m^{\ell}(n) = \begin{cases} -2e_m(n)\varphi'(\mathrm{net}_m^{\ell}(n)), & \ell = K \\ \varphi'\left(\mathrm{net}_m^{\ell}(n)\right) \cdot \sum_{s=1}^{S_{\ell+1}} \boldsymbol{\delta}_s^{\ell+1}(n) \cdot \mathbf{w}_{m,s}^{\ell+1}, & 1 \leq \ell \leq K-1 \end{cases}$$

where K denotes the number of layers, $e_m(n)$ is the error at output node, n denotes the discrete time index, $\varphi'(\cdot)$ is the derivative of the activation function with respect to its input, S_ℓ is the number of inputs in the next layer, and

$$\boldsymbol{\delta}_s^{\ell}(n) = \left[\delta_s^{\ell}(n), \delta_s^{\ell}(n+1), \ldots, \delta_s^{\ell}(n+q^{\ell})\right] \tag{16}$$

is a vector of propagated gradient terms. Each term $\boldsymbol{\delta}_s^{\ell+1}(n) \cdot \mathbf{w}_{m,s}^{\ell+1}$ within the sum corresponds to a reverse FIR filter. That is, delta terms $\delta_s^{\ell}(n)$ are filtered through FIR filter connections to form the deltas for the previous layer. This process is applied layer by layer working backward through the network. The weights are adapted online at each increment of time n. Note that unit delay operators z^{-1} have been replaced with unit advances z^{+1}. Details on the implementation of this noncausal system are addressed in [25].

IV. Proposed Multiresolution FIR Neural-Network-Based Learning Algorithm

In this section, a multiresolution FIR neural-network (MFIRNN)-based learning algorithm is proposed (Fig. 6). The weights and gains are estimated at each level of resolution. The filters $\{\tilde{h}_{j,l}\}$ and $\{\tilde{g}_{j,l}\}$ are used to compute the inverse MODWT (see Section II), where $c_{j+1,n}$ denotes the scaling coefficients and $d_{j+1,n}$ denotes the wavelet coefficients. Note that the same FIR neural-network architecture with different estimated weights and gains is used at each level of resolution. As stated in [16], the first learning (learning scaling coefficients $c_{j,n}$) starts with randomly initialized connection weights and gains, and each subsequent learning (wavelet plus scaling coefficients) starts with the connection weights and gains resulting from previous learning. FIR neural networks provide a time-dependent mapping (the FIR network introduces time delays into the synaptic structure of the network and adjusts their values during the learning phase [10], [25]), making them suitable for time-series prediction. Furthermore, an algorithm is derived for adapting the gain of the activation function at each level of resolution based on the gradient descent method. The use of adaptive gains of the activation functions in MLP and FIR neural networks greatly increases learning speed and produces a good generalization performance on unseen data (see, e.g., [3], [12], and [24]). Note that the algorithm for adapting the gain of the activation function derived here differs from previous reported approaches [12], [17] in the following aspects. First, the proposed adaptive gain algorithm is based on the temporal backpropagation algorithm [25], while the approach reported in [12] is based on the standard backpropagation algorithm. Second, the weights in the proposed adaptive gain algorithm are adapted at each level of resolution and for every training step, while the approach reported in [17] based on the standard backpropagation algorithm adapts the weights using the same gain value of the activation functions at each resolution level training data. Also note that the proposed algorithm uses two activation functions for modeling scaling and wavelet coefficients (see Sections IV-B and IV-C). The proposed MFIRNN-based learning algorithm is suitable for capturing low- and high-frequency information as well as the dynamics of time-varying signals and, as stated in the multiresolution learning paradigm presented in [16], it reveals the correlation structure (low- and high-frequency components) at each level of resolution that may be obscured in the original signal.

A. Temporal Backpropagation With Adaptive Gain Parameter

As stated in Section II, the MODWT pyramid algorithm generates the wavelet coefficients $\{d_{j,n}\}$ and the scaling coefficients $\{c_{j,n}\}$ by cascading convolutions with wavelet $\{\tilde{h}_l\}$ and

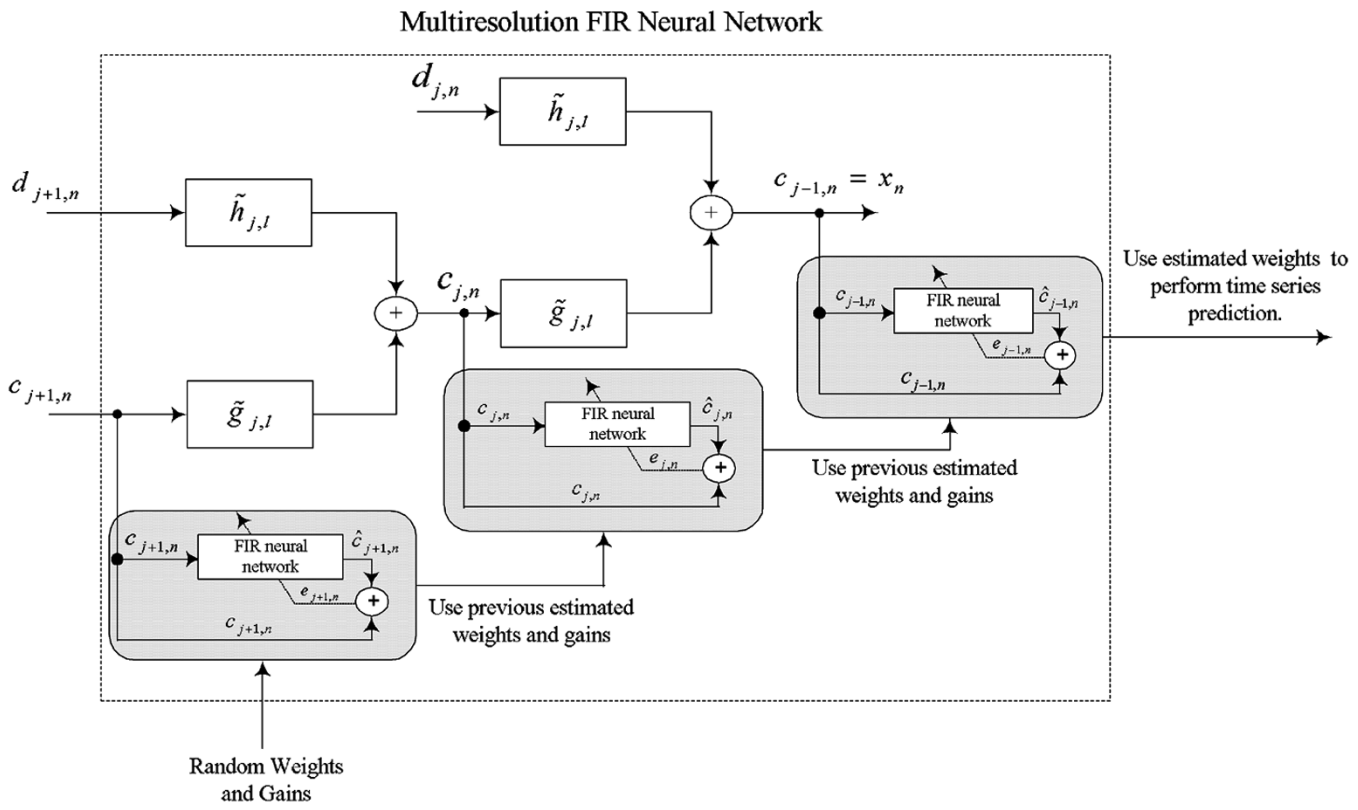

Fig. 6. Architecture of proposed multiresolution FIR NN-based learning algorithm with an adaptive gain in the activation functions at each level of resolution j.

scaling filters $\{\tilde{g}_l\}$ (4). It is worth noting that the weights obtained by the temporal backpropagation algorithm are computed assuming that the gain parameter $\beta_m^\ell = 1$ (15). As a result, the algorithm may fail to converge as the algorithm may be seized in a local minimum of the error surface. To include the gain parameter of activation functions into the temporal backpropagation algorithm at each level of resolution j, we use (15) and (16) to obtain

$$\mathbf{w}_{i,m,j}^\ell(n+1) = \mathbf{w}_{i,m,j}^\ell(n) - \eta_w \delta_{m,j}^\ell(n) \cdot \mathbf{x}_{i,j}^{\ell-1}(n) \quad (17)$$

for $j = 1, 2, \ldots, J$ with the equation

$$\delta_{m,j}^\ell(n) = \begin{cases} -2e_{m,j}(n)\varphi'(\mathrm{net}_{m,j}^\ell(n))\beta_{m,j}^\ell(n), \\ \ell = K \\ \beta_{m,j}^\ell(n)\varphi'\left(\mathrm{net}_{m,j}^\ell(n)\right)\sum_s \boldsymbol{\delta}_{s,j}^{\ell+1}(n)\mathbf{w}_{m,s,j}^{\ell+1}, \\ 1 \le \ell \le K-1 \end{cases}$$

where $\varphi'(\cdot)$ denotes the derivative of the activation function with respect to its input and $\boldsymbol{\delta}_{s,j}^\ell(n) = [\delta_{s,j}^\ell(n), \delta_{s,j}^\ell(n+1), \ldots, \delta_{s,j}^\ell(n+q^\ell)]$ is a vector of propagated gradient terms at each level of resolution j. Equation (17) indicates the extent to which the gain of the activation function in the hidden node participates in learning. The gain can be seen as modulating the learning rate of the weight change, amplifying learning in nodes that find successful directions in weight space. That is, if the gain of the activation function is adjusted, the weight learning rate is actually changed. Note that the weights are adapted online at different levels of resolution j. The selection of the number of level of resolution j depends primarily on the time series under analysis and the wavelet itself. Therefore, the number of levels of resolution can be chosen by analyzing the frequency response magnitude of wavelet coefficients (see, e.g., [21] and [33]).

In the work reported in this paper, the logistic (18) and the hyperbolic tangent (26) functions are considered for modeling the scaling and wavelet coefficients. The logistic function performs a smooth mapping $(-\infty, +\infty) \mapsto (0, \gamma)$, while the hyperbolic tangent function perform a smooth mapping $(-\infty, +\infty) \mapsto (-\gamma, \gamma)$, where γ denotes the slope. Note that the slope and the gain are identical for activation functions with $\gamma = 1$ [24]. The gain of an activation function of a node in a neural network is a constant that amplifies or attenuates the net input to the node.

B. Learning Scaling Coefficients

Assuming that $E\{x_n\} > 0$, the scaling coefficients $\{c_{j,n}\}$ are always non-negative because

$$E\{c_{j,n}\} = E\left\{\sum_{l=0}^{L_j-1} \tilde{g}_{j,l} x_{n-l \bmod N}\right\} = \sum_{l=0}^{L_j-1} \tilde{g}_{j,l} E\{x_{n-l \bmod N}\}$$

since $\sum_{l=0}^{L-1} \tilde{g}_l = 1$, then it follows that (Section II)

$$E\{c_{j,n}\} = \mu_x E\left\{\sum_{l=0}^{L_j-1} \tilde{g}_{j,l}\right\} = \mu_x$$

where μ_x denotes the mean of raw measurements. Thus, the logistic function

$$x_m^\ell(n) = \varphi_{lf}(\mathrm{net}_m^\ell(n)) = \frac{\gamma}{1 + \exp\left(-\beta_{m,n}^k \mathrm{net}_{m,n}^k\right)} \in (0, \gamma) \quad (18)$$

can be used to model the scaling coefficients. The partial derivative of the logistic function with respect to $\beta_m^\ell(n)$ is given by

$$\frac{\partial\varphi_{lf}\left(\text{net}_m^\ell(n)\right)}{\partial\beta_m^\ell(n)} = \frac{\text{net}_m^\ell(n)\exp\left(-\beta_m^\ell(n)\text{net}_m^\ell(n)\right)}{\left[1+\exp\left(-\beta_m^\ell(n)\text{net}_m^\ell(n)\right)\right]^2}. \quad (19)$$

Then, by using the gradient-descent method, an iterative procedure can be derived for updating the gain $\beta_m^\ell(n)$ of the logistic function for learning scaling coefficients at resolution j. Note that a similar procedure for updating γ can also be derived, but we will concentrate only on adaptation of β. By using the gradient descent method, the gain $\beta_{m,j}^\ell(n)$ of the logistic function is thus updated at each increment of time n according to

$$\beta_{m,j}^\ell(n+1) = \beta_{m,j}^\ell(n) - \eta_{lf}\frac{\partial\xi}{\partial\beta_{m,j}^\ell(n)} \quad (20)$$

for $j = 1,2,\ldots,J$, then by using the total cost of the true gradient

$$\frac{\partial\xi}{\partial\beta_{m,j}^\ell(n)} = \sum_{n=1}^{N_T}\frac{\partial\xi}{\partial x_{m,j}^\ell(n)}\cdot\frac{\partial x_{m,j}^\ell(n)}{\partial\beta_{m,j}^\ell(n)} \quad (21)$$

where N_T denotes the number of points in the training sequence. This yields

$$\beta_{m,j}^\ell(n+1) = \beta_{m,j}^\ell(n) - \eta_{lf}\frac{\partial\xi}{\partial x_{m,j}^\ell(n)}\cdot\frac{\partial x_{m,j}^\ell(n)}{\partial\beta_{m,j}^\ell(n)} \quad (22)$$

for $j = 1,2,\ldots,J$, where

$$\frac{\partial\xi}{\partial x_{m,j}^\ell(n)} = \zeta_{m,j}^\ell(n), \qquad \text{for } \ell = K$$

$$= -2e_{m,j}(n)\cdot\frac{\partial\varphi_{lf}\left(\text{net}_{m,j}^\ell(n)\right)}{\partial\beta_{m,j}^\ell(n)} \quad (23)$$

where K denotes the number of layers in the neural network and $e_{m,j}(n)$ is the error at output node at resolution j. For the hidden layers, we obtain

$$\zeta_{m,j}^\ell(n) = \frac{\partial\varphi_{lf}\left(\text{net}_{m,j}^\ell(n)\right)}{\partial\beta_{m,j}^\ell(n)}\cdot\sum_{s=1}^{S_\ell}\boldsymbol{\zeta}_{s,j}^{\ell+1}(n)\cdot\mathbf{w}_{m,s,j}^{\ell+1} \quad (24)$$

for $1\le\ell\le K-1$, where S_ℓ is the number of inputs in the next layer and $\boldsymbol{\delta}_{s,j}^\ell(n) = [\delta_{s,j}^\ell(n), \delta_{s,j}^\ell(n+1),\ldots,\delta_{s,j}^\ell(n+q^\ell)]$ is a vector of propagated gradient terms.

Algorithm 1 (Learning Scaling Coefficients): Using (20)–(24), the following iterative algorithm for updating the gain of the logistic function (18) for learning scaling coefficients at resolution j is then generated

$$\beta_{m,j}^\ell(n+1) = \beta_{m,j}^\ell(n) - \eta_{lf}\zeta_{m,j}^\ell(n)\cdot\text{net}_{m,j}^\ell(n) \quad (25)$$

for $j = 1,2,\ldots,J$, with

$$\zeta_{m,j}^\ell(n) = \begin{cases} \frac{-2e_{m,j}(n)\partial\varphi_{lf}\left(\text{net}_{m,j}^\ell(n)\right)}{\partial\beta_{m,j}^\ell(n)}, \\ \ell = K \\ \frac{\partial\varphi_{lf}\left(\text{net}_{m,j}^\ell(n)\right)\sum_s\boldsymbol{\zeta}_{s,j}^{\ell+1}(n)\cdot\mathbf{w}_{m,s,j}^{\ell+1}}{\partial\beta_{m,j}^\ell(n)}, \\ 1\le\ell\le K-1 \end{cases}$$

where η_{lf} denotes the learning rate for the gain of the logistic function, and the vector of propagated terms at each level of resolution j is denoted by $\boldsymbol{\zeta}_{s,j}^\ell(n) = [\zeta_{s,j}^\ell(n), \zeta_{s,j}^\ell(n+1),\ldots,\zeta_{s,j}^\ell(n+q^\ell)]$.

C. Learning Wavelet Plus Scaling Coefficients

An iterative procedure has also been derived for updating the gain of the hyperbolic tangent function $\{\varphi_{hf}(\cdot)\}$ for learning wavelet plus wavelet coefficients at resolution j. It is important to note that if we assume that $E\{x_n\} = 0$, the wavelet coefficients $\{d_{j,n}\}$ plus the scaling coefficients $\{c_{j,n}\}$ have zero-mean (sign variations) because

$$\begin{aligned} E\{c_{j-1,n}\} =& E\left\{\sum_{l=0}^{L-1}\tilde{h}_l d_{j,n+2^{j-1}l\bmod N}\right. \\ &\left.+\sum_{l=0}^{L-1}\tilde{g}_l c_{j,n+2^{j-1}l\bmod N}\right\} \\ =& \sum_{l=0}^{L-1}\tilde{h}_l E\left\{d_{j,n+2^{j-1}l\bmod N}\right\} \\ &+\sum_{l=0}^{L-1}\tilde{g}_l\, E\left\{c_{j,n+2^{j-1}l\bmod N}\right\} \\ =& \sum_{l=0}^{L-1}\tilde{h}_l E\left\{d_{j,n+2^{j-1}l\bmod N}\right\} \\ &+\sum_{l=0}^{L-1}\tilde{g}_l\mu_x = 0. \end{aligned}$$

The result follows by using the fact that

$$E\{d_{j,n}\} = \sum_{l\in Z}\tilde{h}_l E\{c_{0,n}\} = \mu_x\sum_{l\in Z}\tilde{h}_l = 0$$

where μ_x denotes the mean of raw measurements. Thus, the hyperbolic tangent function $\{\varphi_{hf}(\cdot)\}$

$$\begin{aligned} x_m^\ell(n) &= \varphi_{hf}\left(\text{net}_m^\ell(n)\right) \\ &= \gamma\left[\frac{1-\exp\left(-\beta_m^\ell(n)\text{net}_m^\ell(n)\right)}{1+\exp\left(-\beta_m^\ell(n)\text{net}_m^k(n)\right)}\right]\in(-\gamma,\gamma) \end{aligned} \quad (26)$$

can be used to model this space. The partial derivative of the hyperbolic tangent function with respect to $\beta_m^\ell(n)$ is given by

$$\begin{aligned} \frac{\partial\varphi_{hf}\left(\text{net}_m^\ell(n)\right)}{\partial\beta_m^\ell(n)} &= \frac{\text{net}_m^\ell(n)}{\left[1+\exp\left(-\beta_{m,n}^\ell\text{net}_m^\ell(n)\right)\right]^2} \\ &= \frac{\text{net}_m^\ell(n)}{2}\left[1-\varphi_{hf}^2\left(\text{net}_m^\ell(n)\right)\right]. \end{aligned} \quad (27)$$

The same partial derivative is used at each level of resolution j. By using the gradient descent method, an iterative procedure can be derived for updating the gain $\beta_m^\ell(n)$ of the hyperbolic tangent function for learning scaling plus wavelet coefficients at resolution j. The gain of the hyperbolic tangent function is updated at each increment of time n according to (20)–(24).

Algorithm 2 (Learning Wavelet Plus Scaling Coefficients): Using (20)–(24) and (27), the following iterative algorithm for updating the gain of the hyperbolic tangent function for learning wavelet plus scaling coefficients at resolution j is then generated

$$\beta_{m,j}^\ell(n+1) = \beta_{m,j}^\ell(n) - \eta_{hf}\zeta_{m,j}^\ell(n)\cdot\text{net}_{m,j}^\ell(n) \quad (28)$$

for $j = 1,2,\ldots,J$, with the equation

$$\zeta_{m,j}^\ell(n) = \begin{cases} \frac{-2e_{m,j}(n)\partial\varphi_{hf}\left(\text{net}_{m,j}^\ell(n)\right)}{\partial\beta_{m,j}^\ell(n)} \\ \ell = K \\ \frac{\partial\varphi_{hf}\left(\text{net}_{m,j}^\ell(n)\right)\sum_s\boldsymbol{\zeta}_{s,j}^{\ell+1}(n)\cdot\mathbf{w}_{m,s,j}^{\ell+1}}{\partial\beta_{m,j}^\ell(n)} \\ 1\le\ell\le K-1 \end{cases}$$

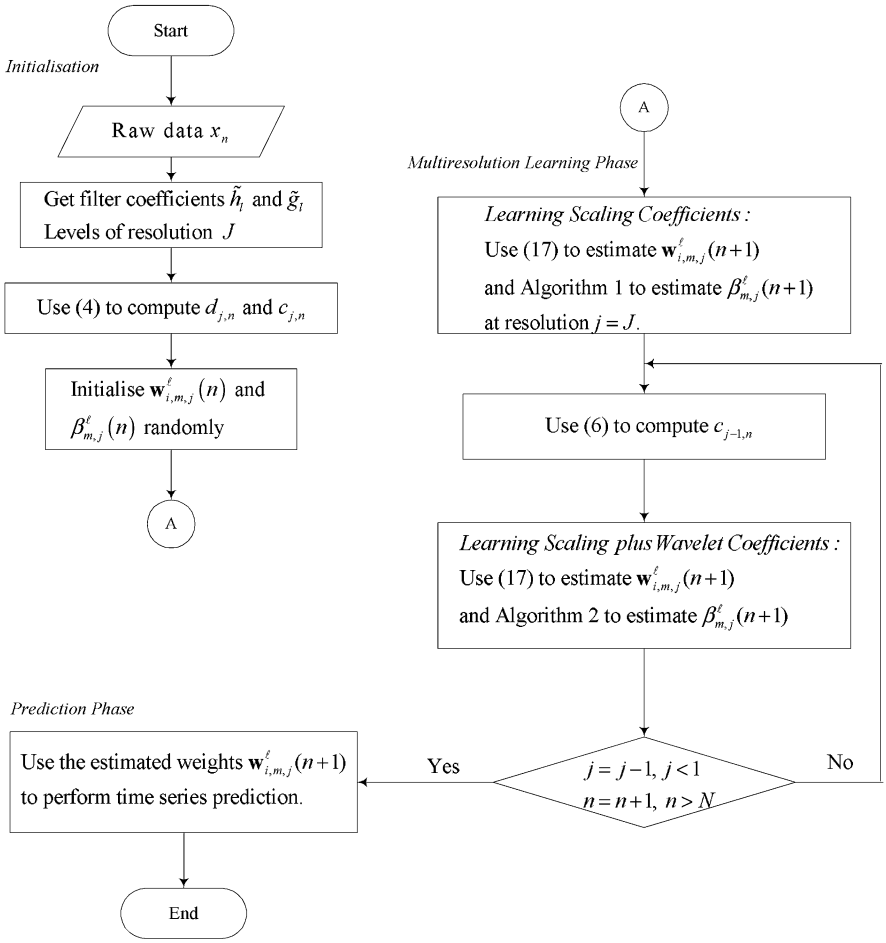

Fig. 7. Flowchart of multiresolution FIR NN-based learning algorithm using the maximal overlap discrete wavelet transform.

where η_{hf} denotes the learning rate for the gain of the hyperbolic tangent function and $\boldsymbol{\zeta}^{\ell}_{s,j}(n) = [\zeta^{\ell}_{s,j}(n), \zeta^{\ell}_{s,j}(n+1), \ldots, \zeta^{\ell}_{s,j}(n+q^{\ell})]$ is a vector of propagated gradient terms at each level of resolution j.

D. Summary of Multiresolution FIR NN-Based Learning Algorithm

The multiresolution FIR neural-network-based learning algorithm here proposed undertakes the problem of learning $\{x_n\}$ by starting from a much simplified version of $\{x_n\}$ using algorithms 1 and 2. These algorithms are used to learn the internal correlation structure of the original signal in the correponding spaces, namely, the approximation space (scaling coefficients) and the residual space (wavelet coefficients). The corresponding learning rates, denoted by $\eta_w, \eta_{hf}, \eta_{lf}$ (see (17), (25), and (28)), for each of the adaptive parameters weights $\{\mathbf{w}^{\ell}_{i,m,j}\}$ and gain of activation functions $\{\beta^{\ell}_{m,j}\}$ can be independently selected. Recall that the slope (steepness) $\gamma\beta$ and the gain β are identical for activation functions with $\gamma = 1$ [24]. A relationship between the learning rate η in the learning algorithm and the gain β of the nonlinear function for FIR neural networks is provided in [34]. In [34], it is shown that changing the gain of the nonlinear function is equivalent to changing the learning and the weights. Such relationships reduce the degrees of freedom and, therefore, simplify the learning procedure by eliminating one of its parameters (see, e.g., [19]).

Algorithm 3 (Multiresolution FIR NN-Based Learning Algorithm): Using algorithms 1 and 2, the following multiresolution learning algorithm is obtained based on the maximal overlap discrete wavelet transform (see also Fig. 7).

1) Initialization:

Step 1) Select the wavelet $\{\tilde{h}_l\}$ and scaling $\{\tilde{g}_l\}$ filters, and the number of levels of resolution J. Recall from Section IV-A that the number of levels of resolution j can be chosen by analyzing the frequency-response magnitude of wavelet coefficients (see, e.g., [21] and [33]). The wavelet and scaling filters must be chosen according to the common features of the events present in real signals.

Step 2) Use (4) to compute the wavelet coefficients $\{d_{j,n}\}$ and the scaling coefficients $\{c_{j,n}\}$ from the training data $\{x_n\}$.

Step 3) Initialize the weights $\{\mathbf{w}^{\ell}_{i,m,j}(n)\}$ and the gains $\{\beta^{\ell}_{m,j}(n)\}$ of the activation functions $\{\varphi_{hf}(\cdot), \varphi_{lf}(\cdot)\}$ randomly.

2) Multiresolution Learning Phase:

Step 4) Present an input set from the learning examples (input-ouput patterns) and compute

the actual output using the actual parameter values. That is, use the scaling coefficients $\{c_{J,n}\}$ at the lowest resolution $j = J$ to estimate the weights $\{\mathbf{w}_{i,m,j}^{\ell}(n+1)\}$ with an adaptive gain parameter $\beta_{m,j}^{\ell}(n+1)$ of the logistic function using (17) and Algorithm 1.

Step 5) Present another input–output pattern from the next learning example and use the following (see Section II)

$$c_{j-1,n} = \sum_{l=0}^{L-1} \tilde{h}_l d_{j,n+2^{j-1}l \bmod N} + \sum_{l=0}^{L-1} \tilde{g}_l c_{j,n+2^{j-1}l \bmod N}$$

to add details to the approximation process. That is, add the wavelet coefficients $\{d_{j,n}\}$ to the scaling coefficients $\{c_{j,n}\}$ at resolution $j = J-1$, and use the previous estimated weights in step 4. Use the iterative algorithms given by (17) and Algorithm 2 to estimate the weights $\{\mathbf{w}_{i,m,j}^{\ell}(n+1)\}$ with an adaptive gain parameter $\{\beta_{m,j}^{\ell}(n+1)\}$ of the hyperbolic tangent function.

Step 6) Go back to step 5. Repeat according to the number of resolutions J.

3) Prediction Phase:

Step 7) Use the estimated weights $\mathbf{w}_{i,m,j}^{\ell}(n+1)$ to perform time-series prediction.

Note that in Step 5, there is only an information increment because only details (wavelet coefficients) are added to the approximation process. This information increment requires the network to learn only the incremental details and, therefore, refining the neural network's learning behavior. Note that in order to increase the learning speed and produce a good generalization performance on unseen data, adaptive activation functions at each level of resolution j have also been used. As stated in [16], the degree of generalization is influenced by how well the correlation structure is learned by the neural network at each level of resolution. Further details on generalization in neural networks can be found in [14].

V. Experimental Results

The underlying aim of the results here reported are twofold: 1) To assess the generalization ability of FIR neural networks [25] and the proposed multiresolution FIR neural-network (MFIRNN)-based learning algorithm (when applied to Ethernet network traffic). 2) To compare with previous reported multiresolution approaches [16], [17] using the same benchmark Ethernet network traffic, a comparison with previously proposed algorithms is also reported. Real-world Ethernet traffic data are used to assess the performance of the proposed MFIRNN. The proposed MFIRNN has also been assessed using recent real-world corporate proxy server traffic data (Internet connections from machines within Digital Equipment Corporation) and Internet traces, and the results are equally encouraging and can be found in [34]. These results, together with an anomaly detection classification algorithm, will be reported in a forthcoming paper.

A. Performance Measures

To quantitatively compute the prediction performance, we may use the mean squared error (MSE) or the root MSE (RMSE). In the work reported in this paper, the normalized mean-squared error (NMSE) and the prediction gain R_p [10], [25], [26] are considered to assess the prediction performance.

1) NMSE: In the normalized mean squared error (also known as average relative prediction variance), a standard measure of fit, is given by [26]

$$\text{NMSE} = \frac{1}{\hat{\sigma}^2 N} \sum_{n=1}^{N} (x_n - \hat{x}_n)^2 \tag{29}$$

where x_n is the true value of the sequence, $\hat{x}_n$ is the prediction, and σ^2 is the variance of the true sequence over the prediction duration N. The normalization (division by the estimated variance of the data, $\hat{\sigma}^2$) removes the dependence on the dynamic range of the data. This normalization implies that if the estimated mean of the data is used as predictor, NMSE $= 1$ is obtained. That is, a value of NMSE $= 0$ indicates perfect prediction while a value of 1 corresponds to simply predicting the average.

2) Prediction Gain: The prediction gain R_p [10] is another standard criterion to evaluate the prediction performance, which is given by

$$R_p = 10 \log_{10} \left(\frac{\hat{\sigma}_x^2}{\hat{\sigma}_e^2} \right) \text{ dB} \tag{30}$$

where $\hat{\sigma}_x^2$ denotes the estimated variance of the incoming signal, and $\hat{\sigma}_e^2$ denotes the estimated variance of the prediction error.

B. High-Speed Network Traffic Prediction

In this section, the following predictive models are assessed: FIR neural networks (FIRNN) [25], multiresolution MLP (MMLP) network [16], [17], the proposed multiresolution FIRNN (MFIRNN), and the RLS linear predictor. Single-step and multistep network traffic prediction of real-world aggregate Ethernet traffic data are performed. The primary objective of this experiment is to assess the generalization ability of the FIR neural network using the proposed multiresolution learning algorithm. A comparison with the FIRNN trained by the temporal backpropagation algorithm (single-resolution technique) [25] and a comparison with the multiresolution approach reported in [16] and [17] are performed. We have chosen to compare these two algorithms only, due to the similarity in the architectures.

The real-world Ethernet traffic data series used in this section are part of a data set collected at Bellcore in August 1989. They correspond to one normal hour's worth of traffic, collected every

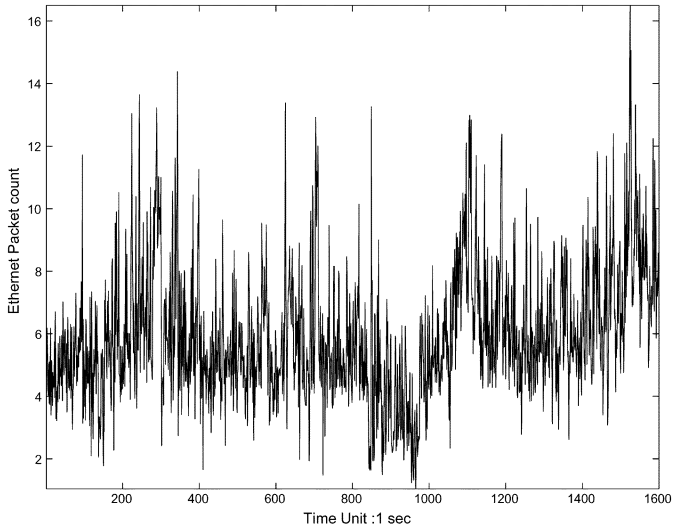

Fig. 8. Ethernet Packet count for time scale 1 second.

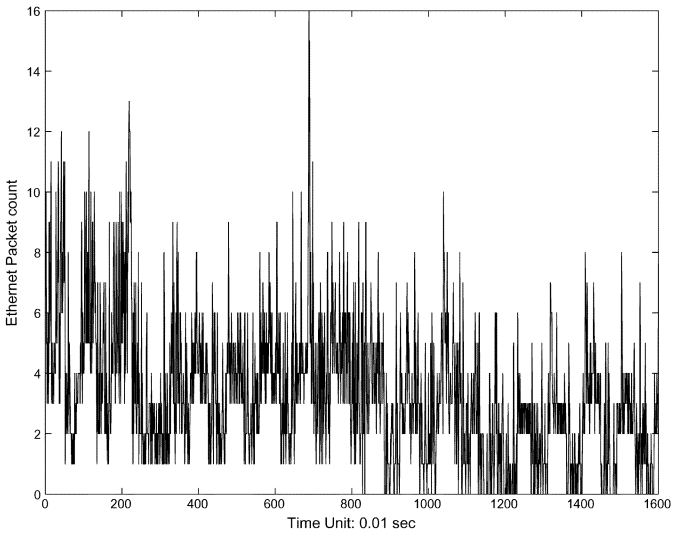

Fig. 9. Ethernet Packet count for time scale 0.01 s.

10 ms, thus resulting in a length of 360 000 samples.[1] Figs. 8 and 9 show an example of the Ethernet traffic in packet counts (i.e., the number of packets per time unit) for two different time scales: 1 s and 0.01 s. The amplitude of the Ethernet traffic data was adjusted to lie in the range of the logistic function (i.e., $\varphi_{lf} \in (0,1)$). The 1000 samples are used for training, and the subsequent 100 data samples following the training data set are used for evaluating the prediction performance.

Table I shows the parameters settings used in each predictor for learning the Ethernet traffic data series. The commonly used notation for feedforward neural networks $1 \times 10 \times 1$ (network dimension) denotes the number of inputs, hidden units, and outputs, respectively. The selection of network dimension and network parameters (i.e., taps per layer) is based on the following

[1]The Ethernet data were collected between August 1989 and February 1992 on several Ethernet LANs at Bellcore Morristown Research and Engineering Center, which carried primarily local traffic, but also all traffic between Bellcore and the Internet [15]. These traces are available at http://ita.ee.lbl.gov/index.html

TABLE I
PARAMETERS SETTINGS FOR LEARNING ETHERNET TRAFFIC DATA SERIES

Model	Network Dimensions	Learning Rate	TDL
FIRNN [25]	$1 \times 10 \times 1$	$\eta = 0.01$	14×1 taps per layer
MMLP [16], [17]	$15 \times 10 \times 1$	$\eta = 0.01$	
RLS [8]		$\lambda = 0.998$	$q = 15$ taps
Proposed MFIRNN	$1 \times 10 \times 1$	$\eta_w, \eta_{hf}, \eta_{lf} = 0.01$	14×1 taps per layer

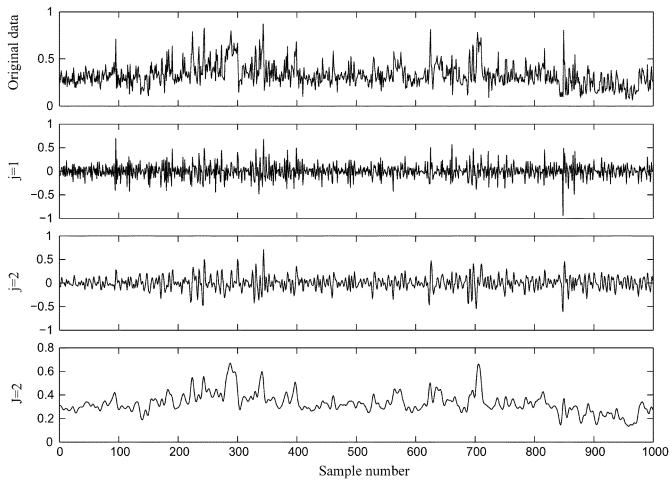

Fig. 10. Ethernet traffic data with two levels of decomposition, $J = 2$, using a quadratic spline wavelet.

heuristic. Since the first layer in the network prior to the sigmoids act as a bank of linear filters, the filter order is obtained by looking at single-step prediction error residuals using linear AR predictors (see e.g., [25]). Based on the analyzed Ethernet traffic data series, the single-step prediction error residuals show a negligible improvement for an order greater than 15. Therefore, the FIR network has equivalent memory of 15 time steps corresponding to the sum of the filter order in each layer $(14+1)$. It is worth noting that the selection of network dimensions for neural networks remains as a difficult problem which requires further research [25], [26]. Some attempts in solving this problem can be found in [14], [37], and references therein. The number of hidden units is chosen based on the following heuristic. We use one hidden layer with ten hidden units, with the number of units approximately equal to half the sum of the number of input and output units (see, e.g., [10] and [38]). The RLS linear predictor is used with 15 taps. The MLP network is also used with 15 inputs and feeding ten neurons in the hidden layer. The conventional logistic activation function is used in the hidden units for the FIRNN and the MMLP network. The number of training epochs for the FIRNN predictor and multiresolution approaches (MMLP and MFIRNN) is set to 100 and $100/(J+1)$, respectively, with $J = 2$. The levels of resolution J are chosen based on the premise that the higher the order of resolutions, the smoother the time series in the residual space and, thus, the less information that the neural network can retrieve. This is illustrated in Fig. 10 (see lower plot, $J = 2$) in which two levels of decomposition for the Ethernet traffic data are shown. The following wavelets are assessed for the proposed MFIRNN: a quadratic spline wavelet, the least-asymmetric (LA) compactly supported wavelets, Daubechies wavelets, and Haar wavelet [6], [18], [21]. In order to achieve a fair comparison, the randomly

TABLE II
SINGLE-STEP NMSE AND R_p FOR THE ETHERNET DATA SERIES USING 1000 SAMPLES FOR TRAINING

Model	Wavelet	NMSE	R_p[dB]
FIRNN [25]		0.6666	2.43
RLS Linear Predictor [8]		0.4613	3.46
MMLP [16], [17]	Haar	0.5775	2.79
Proposed MFIRNN	D(4)	0.2531	6.38
	D(6)	0.2175	7.07
	LA(8)	0.2416	6.58
	Haar	0.2263	6.66
	Spline	0.2521	6.42

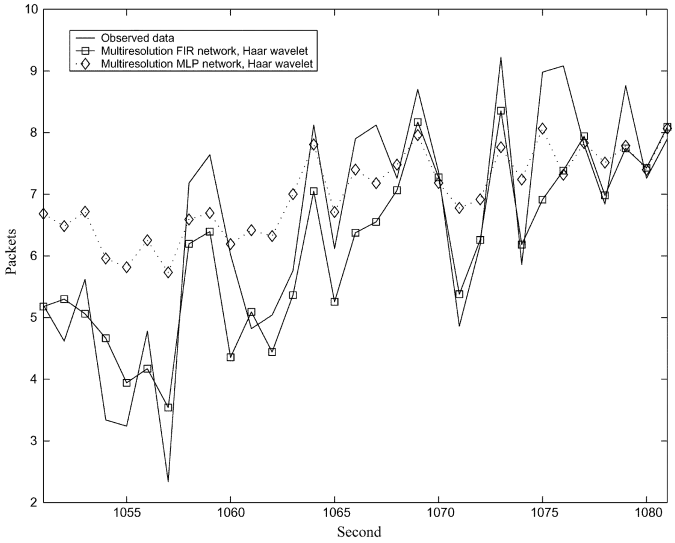

Fig. 11. Single-step prediction for multiresolution FIR network and multiresolution MLP network.

generated initial weights for the networks are identical as are the learning rates for the temporal and backpropagation algorithms.

Table II shows the single-step NMSE and R_p for different predictors. It can be seen that the MFIRNN outperforms the RLS linear predictor and previous reported algorithms (FIRNN and MMLP) using, for example, the Haar wavelet and Daubechies' wavelet with six filter coefficients. These results also indicate the difficulty of predicting high-speed network traffic. Note that the Haar and D(6) wavelets yield better predictions than those resulting from the quadratic spline wavelet. This is because the Haar and D(6) wavelets remove more details of the signal at lower levels of resolution. Therefore, a more simplified version of the signal is learned when using these wavelets. Note that several runs were performed and similar results were obtained in all runs. The results for single-step prediction are shown in Fig. 11 for the MFIRNN and MMLP using the Haar wavelet. Fig. 12 illustrates the results for MFIRNN, FIRNN, and RLS predictor.

To further evaluate the prediction performance of the MFIRNN, we have assessed the MFIRNN and the MMLP using the Haar wavelet for different network structures. Note that in the previous experiment, the Haar and D(6) wavelets yield better predictions than those resulting from the quadratic spline wavelet. Therefore, in this experiment, we have chosen the Haar wavelet only to achieve a fair comparison. The same learning rates shown in Table I are used in this experiment. The

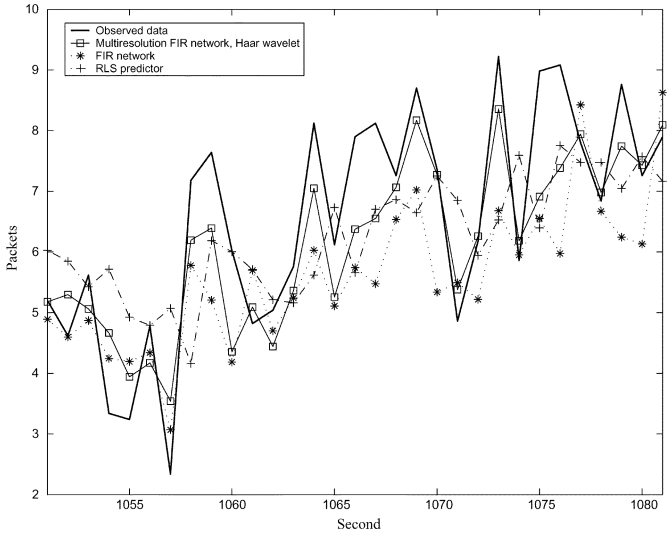

Fig. 12. Single-step prediction for multiresolution FIR network, FIR network, and RLS predictor.

number of training epochs for both multiresolution approaches (MMLP and MFIRNN) is set to $100/(J+1)$ with $J=2$. In this assessment, 1000, 512, and 300 samples (reduced training data set) are used for training, and the subsequent 100 data samples following the training data are used for evaluating the prediction performance. Table III shows the single-step NMSE and R_p for the MFIRNN and the MMLP networks using 1000 training data samples. It can be seen that the MFIRNN outperforms the MMLP using the Haar wavelet for different network structures. Note that the results are quite encouraging since the number of training epochs for each level of resolution is approximately 30 epochs. Tables IV and V show the results for both networks when using 512 and 300 training data samples. It can be seen that there is still an improvement of the MFIRNN over the MMLP for different network structures when using a reduced data set. Note that the best performance of the MFIRNN is achieved with small networks dimensions when using 1000, 512, and 300 training data samples; however, these small networks may not generalize well [14] (Fig. 14). In the next experiment, the generalization ability of the proposed MFIRNN is assessed using different network structures.

To illustrate the generalization ability of the FIR neural network employing the proposed multiresolution learning algorithm, we now turn to multistep prediction. In multistep predictions, the predicted output is fed back as input for the next prediction and all other network inputs are shifted back one time unit. Thus, as the iterated multistep prediction process proceeds, the inputs will eventually consist of entirely predicted values [16]. The same learning rates, network dimensions, and training parameters shown in Table I are used in this experiment. In this assessment, the number of training epochs for the FIRNN predictor and multiresolution approaches (MMLP and MFIRNN) is set to 3000 and $3000/(J+1)$, respectively, with $J=2$. The iterated multistep prediction NMSE for 20 time steps is shown in Fig. 13 for the FIR network, the MMLP, the RLS linear predictor, and the proposed MFIRNN. It can be seen that the MFIRNN prediction outperforms the RLS predictor and previous reported algorithms [16], [17], [25]. Fig. 14

TABLE III
SINGLE-STEP NMSE AND R_p FOR THE MFIRNN AND THE MMLP USING 1000 SAMPLES FOR TRAINING

Model	Network Dimensions	TDL	Wavelet	NMSE	R_p[dB]
MMLP [16], [17]	$7 \times 10 \times 1$		Haar	0.6015	2.21
	$10 \times 10 \times 1$			0.5892	2.44
	$15 \times 10 \times 1$			0.5775	2.79
	$32 \times 10 \times 1$			0.5009	2.98
	$32 \times 20 \times 1$			0.5143	2.84
Proposed MFIRNN	$1 \times 10 \times 1$	6×1 taps per layer	Haar	0.2052	7.90
	$1 \times 10 \times 1$	9×1 taps per layer		0.2457	6.73
	$1 \times 10 \times 1$	14×1 taps per layer		0.2263	6.66
	$1 \times 10 \times 1$	31×1 taps per layer		0.3083	5.08
	$1 \times 20 \times 1$	31×1 taps per layer		0.4451	3.47

TABLE IV
SINGLE-STEP NMSE AND R_p FOR THE MFIRNN AND THE MMLP USING 512 SAMPLES FOR TRAINING

Model	Network Dimensions	TDL	Wavelet	NMSE	R_p[dB]
MMLP [16], [17]	$7 \times 10 \times 1$		Haar	0.8675	1.64
	$10 \times 10 \times 1$			0.7216	1.41
	$15 \times 10 \times 1$			0.7827	1.33
	$32 \times 10 \times 1$			0.8235	1.05
	$32 \times 20 \times 1$			0.8063	1.09
Proposed MFIRNN	$1 \times 10 \times 1$	6×1 taps per layer	Haar	0.3198	5.18
	$1 \times 10 \times 1$	9×1 taps per layer		0.3782	4.33
	$1 \times 10 \times 1$	14×1 taps per layer		0.4401	3.86
	$1 \times 10 \times 1$	31×1 taps per layer		0.6377	2.33
	$1 \times 20 \times 1$	31×1 taps per layer		0.6424	2.23

TABLE V
SINGLE-STEP NMSE AND R_p FOR THE MFIRNN AND THE MMLP USING 300 SAMPLES FOR TRAINING

Model	Network Dimensions	TDL	Wavelet	NMSE	R_p[dB]
MMLP [16], [17]	$7 \times 10 \times 1$		Haar	0.9942	1.15
	$10 \times 10 \times 1$			0.8259	1.27
	$15 \times 10 \times 1$			0.8678	1.12
	$32 \times 10 \times 1$			0.8281	0.79
	$32 \times 20 \times 1$			0.8472	0.81
Proposed MFIRNN	$1 \times 10 \times 1$	6×1 taps per layer	Haar	0.3673	4.30
	$1 \times 10 \times 1$	9×1 taps per layer		0.4494	3.43
	$1 \times 10 \times 1$	14×1 taps per layer		0.4631	3.39
	$1 \times 10 \times 1$	31×1 taps per layer		0.6504	1.88
	$1 \times 20 \times 1$	31×1 taps per layer		0.6552	1.84

shows the results for different network dimensions of the proposed MFIRNN using a Haar wavelet. It can be seen that the network structure $1 \times 10 \times 1$ with 14×1 taps per layer generalizes better than smaller networks (see, e.g., [14]). Thus, for the data set and training parameters used in these results, the generalization ability of the FIR neural network is improved by the proposed multiresolution learning algorithm.

C. Computational Complexity

It is well known that the RLS algorithm requires $\mathcal{O}(q^2)$ operations [8], where q denotes the order of the filter. The proposed multiresolution learning algorithm for FIR networks based on the MODWT uses the same FIR neural network (FIRNN) at each level of resolution. That is, if we assume that there is a simplified FIRNN (e.g., two cascaded linear FIR filters), the first filter is of order q whereas the second filter contains four taps delays. Then, the temporal backpropagation algorithm for FIR networks requires $q+4$ multiplications at each update, that is, the sum of the two filters orders. In general, each neuron in the FIRNN requires on the order of $2ST$ multiplications, where S denotes the nodes per layer and T is the number of taps per layer [25]. Therefore, and since the MODWT requires $\mathcal{O}(N \log_2 N)$ multiplications [21], where N denotes the sample size, the computational complexity involving the MODWT and the FIRNN (i.e., MFIRNN) is approximately $\mathcal{O}((N \log_2 N) + 2ST)$ multiplications. There is thus an increase in computational complexity when using the MODWT, but its computational burden is the same as the widely used fast Fourier transform and, hence, is quite acceptable [21]. It is worth mentioning that the computational complexity of the DWT and the backpropagation algorithm [38] used in the multiresolution learning paradigm (MMLP) reported in [16] are $\mathcal{O}(N)$ multiplications and $\mathcal{O}(W)$ operations, respectively, where W denotes the total number of weights and biases.

VI. CONCLUSION

The currently published learning algorithm for FIR neural networks is single-resolution learning techniques since only one

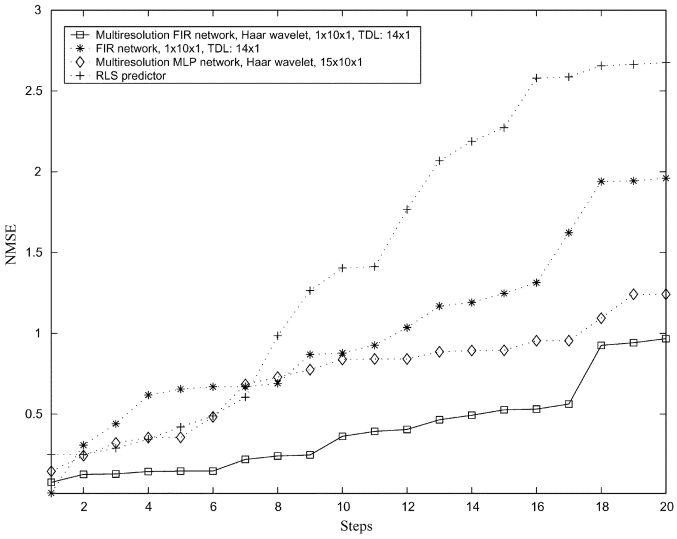

Fig. 13. Iterated multistep prediction error for the Ethernet data series.

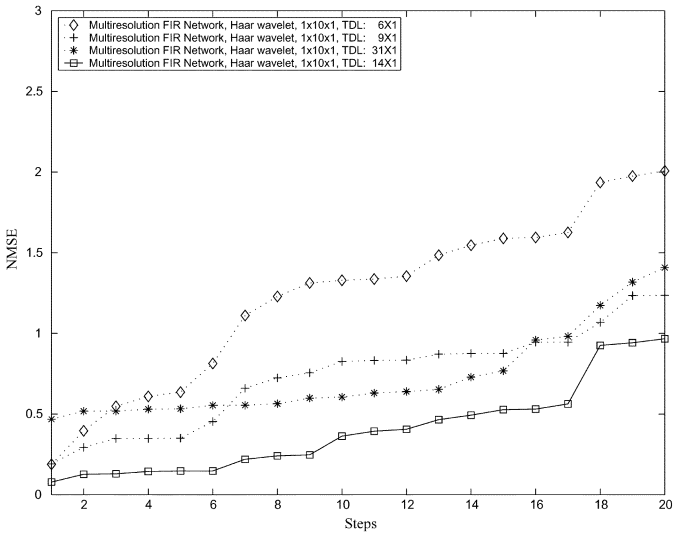

Fig. 14. Iterated multistep prediction error for the Ethernet data series using different network structures of the proposed MFIRNN.

model is used to characterize the measured process. As a result, the network cannot learn complex time series and, hence, the trained network cannot generalize well. In this paper, a multiresolution FIR neural-network-based learning algorithm has been proposed based on the maximal overlap discrete wavelet transform. The proposed multiresolution FIR neural-network learning algorithm, which has its foundations on the multiresolution learning paradigm reported in [16], is suitable for capturing low- and high-frequency information as well as the dynamics of time-varying signals and it has been applied to network traffic prediction. The algorithm employs the multiresolution analysis framework of wavelet theory, which decomposes a signal into wavelet coefficients and scaling coefficients. Experiments are performed to evaluate the performance of the proposed approach using a benchmark Ethernet network traffic. For the algorithms settings and set of test series carried out in this paper, the evidence suggest that the proposed approach can outperform the RLS linear predictor and previous reported algorithms (FIR neural network [25] and multiresolution MLP neural network [16], [17]). Similar encouraging results for recent real-world corporate proxy server traffic data and Internet traces can be found in [34]. Furthermore, the results also suggest that the generalization ability of FIR neural networks can be improved by the proposed multiresolution learning algorithm. At present, further tests are being carried out to produce more experimental evidence of the generalization features of the proposed solution. It is worth noting that there is an increase in the computational complexity of the proposed MFIRNN when compared with the FIRNN [25] and the MMLP [16], [17]; however, this increase is due to the use of the MODWT in the proposed MFIRNN. The computational burden of the MODWT is the same as the widely used fast Fourier transform and, hence, is quite acceptable [21]. Further work will focus on investigating further enhancements using recurrent neural networks and different types of internal memory.

ACKNOWLEDGMENT

The authors would like to thank the referees for their most helpful and constructive comments.

REFERENCES

[1] A. D. Back and A. C. Tsoi, "FIR and IIR synapses, a new neural network architecture for time series modeling," *Neural Comput.*, vol. 3, pp. 375–385, 1991.
[2] B. R. Bakshi, "Multiscale analysis and modeling using wavelets," *J. Chemometrics*, vol. 13, pp. 415–434, 1999.
[3] A. N. Birkett and R. A. Gouban, "Nonlinear adaptive filtering with FIR synapses and adaptive activation functions," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, 1997, pp. 3321–3324.
[4] G. E. P. Box and G. M. Jenkins, *Time Series Analysis: Forecasting and Control*. San Francisco, CA: Holden-Day, 1976.
[5] L. Cao, "Support vector machines experts for time series forecasting," *Neurocomput.*, vol. 51, pp. 321–339, 2003.
[6] I. Daubechies, *Ten Lectures on Wavelets*. New York: SIAM, 1992.
[7] M. Hallas and G. Dorffner, "A comparative study of feedforward and recurrent neural networks in time series prediction," in *Proc. 14th European Meet. Cybernetics Systems Research*, vol. 2, 1998, pp. 644–647.
[8] S. Haykin, *Adaptive Filter Theory*, 3rd ed. Upper Saddle River, NJ: Prentice-Hall, 1996.
[9] ——, "Neural networks expand SP's horizons," *IEEE Signal Process. Mag.*, vol. 13, no. 2, pp. 24–49, Mar. 1996.
[10] ——, *Neural Networks: A Comprehensive Foundation*, 2nd ed. Upper Saddle River, NJ: Prentice-Hall, 1999.
[11] T. Koskela, M. Lehtokangas, J. Saarinen, and K. Kaski, "Time series prediction with multilayer perceptron, FIR and elman neural network," in *Proc. World Congr. Neural Networks*, 1996, pp. 491–496.
[12] J. K. Kruschke and J. R. Movellan, "Benefits of gain: Speeded learning and minimal hidden layers in back-propagation networks," *IEEE Trans. Syst., Man Cybern.*, vol. 21, no. 1, pp. 273–280, Jan./Feb. 1991.
[13] A. Lapedes and R. Farber, "Non-Linear Signal Processing Using Neural Networks: Prediction and System Modeling," Los Alamos National Lab., Tech. Rep. LA-UR-87-2662, Theoretical Div., Los Alamos, NM, 1987.
[14] S. Lawrence, C. Giles, and A. C. Tsoi, "What size neural network gives optimal generalization? convergence properties of backpropagation," Inst. Advanced Comput. Studies, Univ. Maryland, College Park, MD, Tech. Rep. UMIACS-TR-96-22 and CS-TR-3617, 1996.
[15] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of ethernet traffic (extended version)," *IEEE/ACM Trans. Netw.*, vol. 2, no. 1, pp. 1–15, Feb. 1994.
[16] Y. Liang and E. W. Page, "Multiresolution learning paradigm and signal prediction," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2858–2864, Nov. 1997.

[17] Y. Liang, "Adaptive neural activation function in multiresolution learning," in *Proc. IEEE Int. Conf. Systems, Man Cybernetics*, vol. 4, 2000, pp. 2601–2606.
[18] S. Mallat, *A Wavelet Tour of Signal Processing*, 2nd ed. New York: Academic, 1999.
[19] D. Mandic and J. A. Chambers, *Recurrent Neural Networks for Prediction.*. New York: Wiley, 2001.
[20] O. Nerrand, P. Roussel-Ragot, L. Personnaz, and G. Dreyfus, "Neural networks and nonlinear adaptive filtering: Unifying concepts and new algorithms," *Neural Comput.*, vol. 5, pp. 165–199, 1993.
[21] D. B. Percival and A. T. Walden, *Wavelet Methods for Time Series Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 2000.
[22] A. Sang and L. A. San-qi, "Predictability analysis of network traffic," in *Proc. IEEE INFOCOM*, 2000, pp. 342–351.
[23] S. Soltani, "On the use of the wavelet decomposition for time series prediction," *Neurocomput.*, vol. 48, no. 1–4, pp. 267–277, 2002.
[24] G. Thimm, P. Moerland, and E. Fiesler, "The interchangeability of learning rate and gain in back-propagation neural networks," *Neural Comput.*, vol. 8, no. 2, pp. 451–460, 1996.
[25] E. A. Wan, "Finite impulse response neural networks with applications in time series prediction," Ph.D. dissertation, Dept. Elect. Eng., Stanford Univ., Stanford, CA, 1993.
[26] A. S. Weigend and N. A. Gershenfeld, *Time Series Prediction: Forecasting the Future and Understanding the Past*. Reading, MA: Addison-Wesley, 1994.
[27] J. M. Corchado and J. Aiken, "Hybrid artificial intelligence methods in oceanographic forecast models," *IEEE Trans. Syst., Man, Cybern., C, Appl. Rev.*, vol. 32, no. 4, pp. 307–313, Nov. 2002.
[28] B. L. Zhan, R. Coggins, M. A. Jabri, D. Dersh, and B. Flower, "Multiresolution forecasting for futures trading using wavelet decompositions," *IEEE Trans. Neural Netw.*, vol. 12, no. 4, pp. 765–775, Jul. 2001.
[29] F. J. Chang, J. M. Liang, and Y. C. Chen, "Flood forecasting using radial basis function neural networks," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 31, no. 4, pp. 530–535, Nov. 2001.
[30] S. J. Lee and C. L. Hou, "A neural-fuzzy system for congestion control in ATM networks," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 30, no. 1, pp. 2–9, Feb. 2000.
[31] J. Dorronsoro, F. Ginel, C. Sanchez, and C. S. Cruz, "Neural fraud detection in credit card operations," *IEEE Trans. Neural Netw.*, vol. 8, no. 4, pp. 827–834, Jul. 1997.
[32] R. Logeswaran, "A prediction-based neural network scheme for lossless data compression," *IEEE Trans. Syst., Man Cybern. C, Appl. Rev.*, vol. 32, no. 4, pp. 358–365, Nov. 2002.
[33] V. Alarcon-Aquino and J. A. Barria, "Anomaly detection in communication networks using wavelets," *Proc. Inst. Elect. Eng., Commun.*, vol. 148, no. 6, pp. 355–362, 2001.
[34] V. Alarcon-Aquino, "Anomaly detection and prediction in communication networks using wavelet transforms," Ph.D. dissertation, Dept. Elect. Electron. Eng., Imperial College London, Univ. London, U.K., 2003.
[35] R. Sitte and J. Sitte, "Analysis of the predictive ability of time delay neural networks applied to the S&P 500 time series," *IEEE Trans. Syst., Man Cybern. C, Appl. Rev.*, vol. 30, no. 4, pp. 568–572, Aug. 2000.
[36] L. J. Cao and F. E. H. Tay, "Support vector machine with adaptive parameters in financial time series forecasting," *IEEE Trans. Neural Netw.*, vol. 14, no. 6, pp. 1506–1518, Nov. 2003.
[37] J. L. Yuan and T. Fine, "Neural network design for small training sets of high dimensions," *IEEE Trans. Neural Netw.*, vol. 9, no. 2, pp. 266–280, Mar. 1998.
[38] C. Bishop, *Neural Networks for Pattern Recognition*, Oxford, U.K.: Oxford Univ. Press, 1995.

Vicente Alarcon-Aquino (A'94–M'03) was born in Veracruz, Mexico. He received the B.Sc. degree from the Instituto Tecnologico de Veracruz in 1989, the M.Sc. degree from the Instituto Nacional de Astrofísica, Optica y Electrónica, Mexico, in 1993, and the Ph.D. and D.I.C. degrees from Imperial College London, University of London, London, U.K., in 2003, all in electrical engineering.

From 1993 to 1998, he was an Assistant Professor in the Department of Electrical and Electronic Engineering at the Universidad de las Américas, Puebla, Mexico. From 1998 to 2000, he was a Laboratory Demonstrator in the Communications and Signal Processing Laboratory, Imperial College London. Currently, he is an Associate Professor in the Department of Electrical and Electronic Engineering at the Universidad de las Américas, Puebla, Mexico. His research interests include network protocols, network intrusion detection, time-series prediction, wavelet-based digital signal processing, hardware description languages, and multiresolution neural networks. He has authored many technical publications published in conference proceedings and journals, and has been an invited speaker at many national conferences.

Dr. Alarcon-Aquino is a member of technical program committees for several international conferences. He is General Chair of the International Conference on Electronics, Communications and Computers (CONIELECOMP 2005) sponsored by the IEEE Computer Society, and since 2004, belongs to the National System of Researchers (SNI).

Javier A. Barria (M'02) received the B.Sc. degree in electronic engineering from the University of Chile, Santiago, Chile, in 1980 and the Ph.D. and M.B.A. degrees from Imperial College London, London, U.K., in 1992 and 1998, respectively.

From 1981 to 1993, he was a System Engineer and Project Manager (network operations) with the Chilean Telecommunications Company. Currently, he is a Senior Lecturer in the Intelligent Systems and Networks Group, Department of Electrical and Electronic Engineering, Imperial College London. His research interests include the application of intelligent agent systems and distributed algorithms to solve telecommunication network problems. More recently, his interest has focused on communication networks monitoring strategies using signal and image processing techniques, distributed resource allocation in dynamic topology networks, and fair and efficient resource allocation in IP environments. He has published several learned journal papers in the area of communication networks in the design of robust and fault-tolerant communication systems and networks, numerical procedures to solve large-scale systems, and load control and routing strategies. He is the joint holder of several FP5 and FP6 European Union project contracts all concerned with aspects of communication systems design and management.

Dr. Barria is a member of the Institute of Electrical Engineers. He was a British Telecom Research Fellow from 2001 to 2002.