On the Weight Convergence of Elman Networks

Qing Song, *Member, IEEE*

***Abstract*—An Elman network (EN) can be viewed as a feedforward (FF) neural network with an additional set of inputs from the context layer (feedback from the hidden layer). Therefore, instead of the offline backpropagation-through-time (BPTT) algorithm, a standard online (real-time) backpropagation (BP) algorithm, usually called Elman BP (EBP), can be applied for EN training for discrete-time sequence predictions. However, the standard BP training algorithm is not the most suitable for ENs. A low learning rate can improve the training of ENs but can also result in very slow convergence speeds and poor generalization performance, whereas a high learning rate can lead to unstable training in terms of weight divergence. Therefore, an optimal or suboptimal tradeoff between training speed and weight convergence with good generalization capability is desired for ENs. This paper develops a robust extended EBP (eEBP) training algorithm for ENs with a new adaptive dead zone scheme based on eEBP training concepts. The adaptive learning rate and adaptive dead zone optimize the training of ENs for each individual output and improve the generalization performance of the eEBP training. In particular, for the proposed eEBP training algorithm, convergence of the ENs' weights with the adaptive dead zone estimates is proven in the sense of Lyapunov functions. Computer simulations are carried out to demonstrate the improved performance of eEBP for discrete-time sequence predictions.**

***Index Terms*—Elman networks (ENs), extended Elman backpropagation (eEBP) training, generalization, weight convergence.**

I. Introduction

ELMAN NETWORKS (ENs) [1] are able to learn interesting internal representations of discrete-time sequences. In ENs, recurrent feedback takes place in the hidden layer, as opposed to a Jordan network [2], [3], which has an output-layer feedback structure. ENs reveal a rich structure that allows them to be highly context dependent while also expressing generalizations across classes of items. However, to have real-time (online) learning capability, standard backpropagation (BP) training for ENs, known as Elman BP (EBP) [1]–[6], by gradient-descent methods faces severe problems [7], [8]. Designing efficient training algorithms for ENs and recurrent networks remains a challenging research problem [4]–[28]. Furthermore, the generalization ability of recurrent neural networks constitutes an additional, and not yet satisfactorily solved, question. Unlike standard feedforward (FF) networks, common recurrent neural architectures possess a Vapnik–Chervonenkis dimension, which depends on the maximum length of input sequences and is hence, in theory, infinite for arbitrary inputs [9], [13], [14], [29].

Weight convergence is a very important issue for recurrent neural networks because of their inherent signal feedback structure [10], [14], [23], [28]. Kuan *et al.* [10] rigorously analyzed the convergence properties of a BP algorithm for recurrent networks, including EBP training, containing either output or hidden-layer recurrence by assuming that the estimated weight norm is bounded. These conditions permit data to be generated by stochastic processes with considerable dependence. Kuan *et al.* suggested restrictions that might have helped ensure convergence of the network parameters to a local optimum, as some simulations illustrate. However, this approach did not result in a weight convergence that could guarantee a bounded small estimate of the weight norm to improve generalization performance [9].

This paper focuses on the scenario of real-time online learning for ENs of an extended EBP (eEBP) training algorithm with a new multiple-input–multiple-output (MIMO) adaptive dead zone scheme and guaranteed weight convergence. A convergence proof of the eEBP training algorithm is provided and an extended gradient term with the adaptive dead zone is used to automatically obtain necessary information in a real time, noisy environment. Unlike the previous convergence analysis of ENs [10], this paper provides a guaranteed convergence proof of the weight norm for the hidden and output layers, respectively, without assuming boundedness of the estimated weight norm. To the best of our knowledge, no previous research has provided a complete weight convergence proof for ENs-type MIMO adaptive systems with an adaptive dead zone scheme. To guarantee the weight convergence of ENs, the proposed eEBP training of ENs is adaptive to estimate the dead zone variables, which is critical for the adaptive systems to be used under different noisy environments of the MIMO system. This important feature is different from the multiple-input–single-output (MISO) adaptive systems with a fixed dead zone from previous studies [13], [14], [30]. This paper explores the inherent dynamic learning structure of the underlying training algorithm to improve weight convergence and, subsequently, the generalization performance of ENs. This approach also improves the accuracy of eEBP training compared to standard EBP training. An optimal or suboptimal tradeoff is needed between training and testing errors, as in the structural risk minimization (SRM) principle [29], [31]. This research provides an eEBP training algorithm with an adaptive dead zone learning scheme that is similar to the adaptive filtering and prediction system, but with a new nonlinear system extension [18], [30], specifically for ENs. The eEBP training uses the adaptive dead zone to make an optimal or suboptimal tradeoff by shutting down weight

Manuscript received May 12, 2009; revised October 12, 2009; accepted December 09, 2009. First published February 02, 2010; current version published February 26, 2010.

The author is with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798, Singapore (e-mail: eqsong@ntu.edu.sg).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNN.2009.2039226

updating if the training error is below an adaptive threshold value. This adaptive learning approach is different from that in previous research of the fixed dead zone scheme [14], [30] in that eEBP learning can be automatically switched on and off for ENs in an optimal or suboptimal manner, depending on the disturbance levels of each output. This scheme leads to an optimal compromise between training and testing errors, which particularly benefits the weight convergence and generalization performance of ENs. The inherent dynamic learning structure of the underlying training algorithm is examined for improved weight convergence and, consequently, generalization performance. The simulations show that the proposed eEBP is in accord with the basic concepts of manual weight norm limitation algorithms [9], [10], [22], in that the automatically small weight implementation of eEBP leads to better generalization performance than that of EBP.[1]

The rest of this paper is organized as follows. Section II briefly introduces the structure of ENs and the EBP and eEBP training algorithms. Section III introduces a Lyapunov function as the theoretical foundation for the convergence proof of the eEBP algorithm. Section IV presents computer simulation results to demonstrate the efficiency of the proposed eEBP training. Section V draws the study's final conclusions.

II. ENs and EBP/eEBP Training Algorithms

A. EN Structure and Standard EBP Training

Consider the EN structure in Fig. 1: it has an h-dimensional external input vector $y^{(0)}(t) = [y_1^{(0)}(t), y_2^{(0)}(t), \ldots, y_h^{(0)}(t)]^T$ and an h-dimensional output vector $y^{(2)}(t) = [y_1^{(2)}(t), y_2^{(2)}(t), \ldots, y_h^{(2)}(t)]^T$. The structure has an output-layer weight matrix $W^{2,1}(t) \in R^{h\times m}$ and a hidden-layer weight matrix $W^{1,0}(t) = [W^{1,0,I}(t) W^{1,0,H}(t)] \in R^{m\times n}$, with $W^{1,0,I}(t) \in R^{m\times h}$ and $W^{1,0,H}(t) \in R^{m\times m}$, where $W^{1,0,H}(t)$ is the context subweight matrix, $W^{1,0,I}(t)$ is the input signal subweight matrix, and m is the number of hidden-layer neurons with $n = m + h$.

For the input vector $y^{(0)}(t)$ of N samples $\{y^{(0)}(0), y^{(0)}(1), \ldots, y^{(0)}(N)\}$, the corresponding EN's output can be expressed in vector form as

$$y^{(2)}(t) = F^{2,1}(t) = [F_1^{2,1}(t) \cdots F_h^{2,1}(t)]^T \in R^{h\times 1} \quad (1)$$

$$a^{(2)}(t) = W^{2,1}(t) y^{(1)}(t) = [a_1^{(2)}(t) \cdots a_h^{(2)}(t)] \quad (2)$$

$$F_k^{2,1}(t) = f(a_k^{(2)}(t)) = \frac{1}{1+\exp(-4a_k^{(2)}(t))} \quad (3)$$

$$a_k^{(2)}(t) = \sum_{j=1}^{m} W_{kj}^{2,1}(t) F_j^{1,0}(t) \quad (4)$$

[1]A small weight norm does not necessarily imply that in general ENs have good generalization performance; rather, it is because ENs with relatively high learning rates can cause weight divergence and increase the weight norm. This finding, to a certain extent, matches those of [9] that under specific situations a relatively small weight norm is preferred for good generalization (indeed, the simulations in Section IV demonstrate that all weight norms of the ENs in the given examples are monotonically increasing from relatively small initial norms, so a relatively large weight norm may indicate weight divergence).

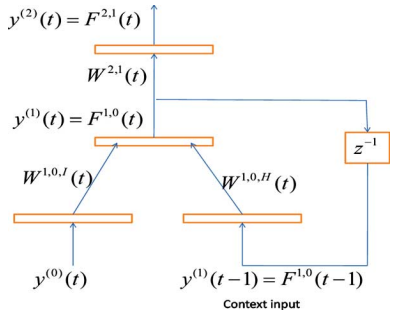

Fig. 1. EN structure where z^{-1} is a one-step time-delay factor.

where $F_k^{2,1}(t)(1 \le k \le h)$ are the output-layer neurons,[2] and $W_{kj}^{2,1}(t)$ represents the components at the kth row and jth column of the output-layer weight matrix $W^{2,1}(t)$. Here $y^{(1)}(t) \in R^{m\times 1}$ is the hidden-layer output vector, defined as

$$y^{(1)}(t) = F^{1,0}(t) = [F_1^{1,0}(t) \quad F_2^{1,0}(t) \quad \cdots \quad F_m^{1,0}(t)]^T \quad (5)$$

$$a^{(1)}(t) = W^{1,0}(t) x(t) = [a_1^{(1)}(t) \quad \cdots \quad a_m^{(1)}(t)] \quad (6)$$

where $F_j^{1,0}(t) = f(a_j^{(1)}(t))\ (1 \le j \le m)$ is the activation function, $W_j^{1,0}(t)$ represents the jth row of the hidden-layer weight matrix $W^{1,0}(t)$, and

$$x(t) = [x_1(t) \quad \cdots \quad x_h(t) x_{h+1}(t) \quad \cdots \quad x_n(t)]^T = [y_1^{(0)}(t) \quad \cdots \quad y_h^{(0)}(t) \quad F_1^{1,0}(t-1) \quad \cdots \quad F_m^{1,0}(t-1)]^T \quad (7)$$

$$= [(y^{(0)}(t))^T \quad (y^{(1)}(t-1))^T]^T \in R^{n\times 1} \quad (8)$$

is the input vector of ENs consisting of the external input $y^{(0)}(t)$ and the one-step delayed hidden-layer state feedback $y^{(1)}(t-1)$, known as the ENs' context input [1].

To predict a desired discrete-time sequence $\{u(0), u(1), \ldots, u(N)\}$, the ENs' input vector $y^{(0)}(t+1)$ is one step ahead of the desired output $u(t)$, that is, $y^{(0)}(t+1) = u(t)$ in the discrete-time sequence prediction problem [1]. Therefore, a bounded disturbance term $\nu(t)$ needs to be introduced to represent the uncertainty in the prediction of the random sequence, since the future desired output is unknown, such that the prediction error is defined as

$$e(t) = u(t) - y^{(2)}(t) + \nu(t) \quad (9)$$

where $\nu(t)$ acts as an external noise input of the total (equivalent) noise level, as used in (35) of the weight convergence proof in Section III.

For a fast online (real-time) training algorithm, the instantaneous squared error criterion is used for the proposed eEBP training algorithm

$$E(t) = \frac{\|e(t)\|^2}{2}. \quad (10)$$

[2]We set a specific gain parameter 4 for the sigma function $f(.)$ in (3) to simplify the weight convergence proof in Section III since the maximum of the specific derivative function $f'(a^{(2)}(t)) = \exp(-4a^{(2)}(t))/1+\exp(-4a^{(2)}(t)) \le 1$. Note that other reasonable positive values can also be used for the gain parameter without affecting the fundamental performance of ENs.

The ENs' training objective can be stated as recursively updating the weights of each layer to map the network output into the given data in an optimal or suboptimal manner so that $E(t)$ is minimized. Specifically, in an environment of time-varying signal statistics, a gradient-type algorithm, such as standard EBP training, can be used to recursively reduce $E(t)$ by estimating the weight at each instant as follows:

$$\begin{aligned} W^{2,1}(t+1) &= W^{2,1}(t) - \epsilon \frac{\partial E(t)}{\partial W^{2,1}(t)} \\ &= W^{2,1}(t) + \epsilon F'^{2,1}(t) e(t) F^{1,0T}(t) \end{aligned} \tag{11}$$

$$\begin{aligned} W^{1,0}(t+1) &= W^{1,0}(t) - \epsilon \frac{\partial E(t)}{\partial W^{1,0}(t)} \\ &= W^{1,0}(t) + \epsilon F'^{1,0}(t) W^{2,1T}(t) F'^{2,1}(t) e(t) x^T(t) \end{aligned} \tag{12}$$

where ϵ is the fixed learning rate of the EBP and the diagonal matrix $F'^{2,1}(t)$ is

$$F'^{2,1}(t) = \operatorname{diag}[\, F_1'^{2,1}(t) \quad F_2'^{2,1}(t) \quad \cdots \quad F_h'^{2,1}(t) \,] \in R^{h \times h} \tag{13}$$

with $F_k'^{2,1}(t) = f'(a_k^{(2)}(t))$ $(1 \le k \le h)$. Also

$$\begin{aligned} &F'^{1,0}(t) \\ &= \operatorname{diag}[\, F_1'^{1,0}(t) \quad F_2'^{1,0}(t) \quad \cdots \quad F_m'^{1,0}(t) \,] \in R^{m \times m} \end{aligned} \tag{14}$$

where $F_j'^{1,0}(t) = f'(a_j^{(1)}(t))$ $(1 \le j \le m)$.

Note that this study focuses on online pattern training instead of offline batch training. Therefore, the discrete-time step t appears explicitly in the iteration (11) and (12), which is similar to Kuan *et al.*'s work [10] but different from the standard weight-updating presentation of EBP for ENs [1]. This paper also uses matrices for weight updating rather than individual weight-updating rules to enable the convergence proof of eEBP training.

B. Proposed eEBP Training Algorithm

The proposed eEBP training algorithm of the ENs is similar to EBP, in that the output- and hidden-layer weights are updated based on two-step iterations. The weight-updating law for the output layer is defined as

$$\begin{aligned} W^{2,1}(t+1) &= W^{2,1}(t) - \frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} \frac{\partial E(t)}{\partial W^{2,1}(t)} \\ &= W^{2,1}(t) + \frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} F'^{2,1}(t) e(t) F^{1,0T}(t) \end{aligned} \tag{15}$$

with the adaptive dead zone estimate vector

$$\Delta^{2,1}(t+1) = \Delta^{2,1}(t) + \frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} \bar{e}(t) \tag{16}$$

where $\bar{e}(t) = [|e_1(t)| \ldots |e_h(t)|]^T$ is a positive vector with absolute values of the components in $e(t)$. Note that the adaptive dead zone is specifically designed for the eEBP training to prevent weight drift and provide guaranteed weight convergence, as proven in Section III [14], [30]. The components of the adaptive dead zone vector $\Delta^{2,1}(t)$ are monotonically increasing until they reach the optimal values (see Section III for more details) and only norm of the adaptive dead zone vector is used for the adaptive learning rate calculation in (17), which is bounded by the optimal or suboptimal value $\Delta^{*2,1}$. The adaptive dead zone can be estimated from iteration (15) to switch the eEBP learning on or off, depending on the estimated noise level, which is done by using $\epsilon^{2,1}(t)$ as the adaptive learning rate

$$\begin{cases} \epsilon^{2,1}(t) = 1, & \text{if } \|e(t)\| \ge \|\Delta^{2,1}(t)\| \\ \epsilon^{2,1}(t) = 0, & \text{if } \|e(t)\| < \|\Delta^{2,1}(t)\| \end{cases} \tag{17}$$

where $\rho^{2,1}(t)$ is a normalization factor defined as

$$\rho^{2,1}(t) = \frac{1 + \|F^{1,0}(t)\|^2 \|F'^{2,1}(t)\|^2}{F_{\min}'^{2,1}} \tag{18}$$

with $F_{\min}'^{2,1} = \min(F_k'^{2,1}(t)) \ne 0$, $\forall k, t$.

The hidden layer of ENs is trained by the following specific gradient algorithm with the extended gradient $(\partial E(t)/\partial x(t))(\partial x(t)/\partial W^{1,0}(t))$, which is not used in the standard EBP hidden-layer training of (12) and is designed to gain extra training information from the recurrent (context) input $F^{1,0}(t-1)$ as in Fig. 1:

$$\begin{aligned} W^{1,0}(t+1) = W^{1,0}(t) &- \frac{\epsilon^{1,0}(t)}{\rho^{1,0}(t)} \left[\frac{\partial E(t)}{\partial W^{1,0}(t)} \right. \\ &\left. + \gamma^{1,0}(t) \frac{\partial E(t)}{\partial x(t)} \frac{\partial x(t)}{\partial W^{1,0}(t)} \right] \end{aligned} \tag{19}$$

$$\Delta^{1,0}(t+1) = \Delta^{1,0}(t) + \frac{\epsilon^{1,0}(t)}{\rho^{1,0}(t)} \|e(t)\| \tag{20}$$

where $\epsilon^{1,0}(t)$ is the adaptive dead zone learning rate of the hidden layer

$$\begin{cases} \epsilon^{1,0}(t) = 1, & \text{if } \|e(t)\| \ge \Delta^{1,0}(t) \\ \epsilon^{1,0}(t) = 0, & \text{if } \|e(t)\| < \Delta^{1,0}(t) \end{cases} \tag{21}$$

with $\Delta^{1,0}(t)$ being an adaptive dead zone parameter, defined according to the weight convergence proof of the hidden layer. The hidden-layer adaptive dead zone estimate is positive [$\Delta^{1,0}(t)$ always has a positive initial value] and monotonically increasing until it reaches an optimal or suboptimal value (see Theorem 2 in Section III).

To guarantee convergence of the hidden-layer training (19) based on the extended gradient, a hybrid control parameter $\gamma^{1,0}(t)$ is specifically defined for eEBP as

$$\begin{cases} \gamma^{1,0}(t) = 1, & \text{if } \beta \ge \Delta_W^{1,0}(t) \ge 0 \\ \gamma^{1,0}(t) = 0, & \text{if } \Delta_W^{1,0}(t) > \beta \text{ or } \Delta_W^{1,0}(t) < 0 \end{cases} \tag{22}$$

where $\gamma^{1,0}(t)$ is designed to prevent the extended gradient $(\partial E(t)/\partial x(t))(\partial x(t)/\partial W^{1,0}(t))$ in (19) from becoming negative or unbounded and guarantee weight convergence (see Theorem 2 in Section III), such that the extended gradient information will not be cut off in these two cases. Here β is a positive constant and $\Delta_W^{1,0}(t)$ is an extended gradient variable defined as shown in (23) at the bottom of the next page, where $\underline{W}^{1,0}(t) = [\, W_1^{1,0}(t) \quad W_2^{1,0}(t) \quad \cdots \quad W_m^{1,0}(t) \,] \in R^{1 \times (m \times n)}$ is a long vector version of the weight matrix $W^{1,0}(t)$, $\bar{x}(t) = [\, \bar{x}_1^T(t) \quad \cdots \quad \bar{x}_n^T(t) \,]^T = \{x(t)x^T(t)\}^{-1} \approx$

$\{\delta I + x(t)x^T(t)\}^{-1}\bar{x}_k(t) \in R^{1\times n}, 1 \le k \le n$, which is similar to [14].[3]

In addition to an adaptive learning rate similar to the output-layer case, this study uses the extended gradient to design a fully adaptive learning algorithm for eEBP instead of the EBP's fixed learning rate. This study uses both standard online BP and real-time recurrent learning (RTRL) [7], [26] methods, depending on the required convergence conditions. The partial derivatives for the extended gradient of the hidden layer are

$$-\frac{\partial E(t)}{\partial x(t)}\frac{\partial x(t)}{\partial W^{1,0}(t)} = F'^{1,0}(t)W^{2,1T}(t)F'^{2,1}(t) \times e(t)\underline{W}^{1,0}(t)\frac{\partial x(t)}{\partial W^{1,0}(t)} \in R^{m\times n} \quad (24)$$

$$-\frac{\partial E(t)}{\partial W^{1,0}(t)} = F'^{1,0}(t)W^{2,1T}(t)F'^{2,1}(t)e(t)x^T(t) \quad (25)$$

where

$$\frac{\partial x(t)}{\partial W^{1,0}(t)} = \left[\frac{\partial x(t)}{\partial W_1^{1,0}(t)} \quad \frac{\partial x(t)}{\partial W_2^{1,0}(t)} \quad \cdots \quad \frac{\partial x(t)}{\partial W_m^{1,0}(t)}\right]^T \in R^{(m\times n)\times n} \quad (26)$$

and the extended recurrent gradient (Jacobian matrix) $(\partial x(t)/\partial W_j^{1,0}(t)) \in R^{n\times n}$ for the hidden layer is given by (27) shown at the bottom of the page.

Note that the approximation of $(\partial F_i^{1,0}(t-1)/\partial W_j^{1,0}(t)) \approx (\partial F_i^{1,0}(t-1)/\partial W_j^{1,0}(t-1))$ $(1 \le i, j \le m)$ is similar to the RTRL scheme for eEBP real-time online training with the assumption of small incremental weight changes for each iteration step [26].

$\rho^{1,0}(t)$ is the normalization factor of the hidden layer

$$\rho^{1,0}(t) = \frac{1}{2F'^{1,0}_{\min}F'^{2,1}_{\min}} \times \left(1+\|F'^{1,0}(t)W^{2,1T}(t)F'^{2,1}(t)\|^2\|x^T(t)+\underline{W}^{1,0}(t)\|^2\right). \quad (28)$$

[3]Similar to a general nonlinear iteration algorithm (see [16, pp. 126–127]), a small positive perturbation constant $\delta > 0$ is used to ensure that $\{\delta I + x(t)x^T(t)\}^{-1}$ is full rank.

Remark 1: The proposed eEBP is different from standard EBP training in several important aspects.

- The adaptive dead zone vector $\Delta^{2,1}(t)$ is specifically designed for the MIMO system, in which each dead zone component is estimated with respect to the individual training error. This feature allows the eEBP to train each of the individual outputs in an optimal or suboptimal manner to minimize noise (see the simulations in Section IV for more details) for the output-layer training, which is different from previous approaches of SISO fixed dead zone training algorithms [13], [14], [30]. In addition, the hidden-layer adaptive dead zone is estimated as a single parameter $\Delta^{1,0}(t)$ that allows the real-time implementation of the adaptive dead zone for the ENs.
- The adaptive learning rates and adaptive dead zone parameters are used together according to the specific weight convergence results of output and hidden-layer training (see Section III), respectively. This adaptive scheme can make an optimal or suboptimal tradeoff between training errors (empirical risk) and testing errors (generalization performance), as opposed to the fixed learning rate of standard EBP, which automatically cuts off the training whenever necessary (see the simulation results in Section IV for more details).
- The extended gradient $(\partial E(t)/\partial x(t))(\partial x(t)/\partial W^{1,0}(t))$ is used in (24), which is designed to gain extra training information from the recurrent (context) input $F^{1,0}(t-1)$. It is part of the input vector $x(t)$, as shown in Fig. 1. Therefore, eEBP is similar to RTRL training [26], to a certain extent, in its ability to capture a sequence's time structure more quickly than standard EBP (see the simulations in Section IV).
- The combination of the adaptive learning rate, the adaptive dead zone, and the hybrid control parameter $\gamma^{1,0}(t)$ guarantees weight convergence for eEBP training. The weight can converge to an optimal or at least suboptimal value (due to the inherent nonlinear dynamics) in the sense of a Lyapunov function, as proven in Section III, and in turn lead to a relatively smaller weight norm for good generalization

$$\begin{aligned}\Delta_W^{1,0}(t) &= \underline{W}^{1,0}(t)\frac{\partial x(t)}{\partial W^{1,0}(t)}\bar{x}(t)x(t)\\ &= \left[0 \quad \cdots \quad 0 \quad \sum_{i=1}^{n}\sum_{l=1}^{m}\left(\frac{\partial F_1^{1,0}(t-1)}{\partial W_{il}^{1,0}(t-1)}W_{il}^{1,0}(t)\right) \quad \cdots \quad \sum_{i=1}^{n}\sum_{l=1}^{m}\left(\frac{\partial F_m^{1,0}(t-1)}{\partial W_{il}^{1,0}(t-1)}W_{il}^{1,0}(t)\right)\right]^T \bar{x}(t)x(t)\\ &= \sum_{j=1}^{m}\left\{\sum_{i=1}^{n}\sum_{l=1}^{m}\left(\frac{\partial F_j^{1,0}(t-1)}{\partial W_{il}^{1,0}(t-1)}W_{il}^{1,0}(t)\right)\bar{x}_{h+j}(t)F_j^{1,0}(t-1)\right\}\end{aligned} \quad (23)$$

$$\begin{aligned}\frac{\partial x(t)}{\partial W_j^{1,0}(t)} &= \left[\frac{\partial u_1(t)}{\partial W_j^{1,0}(t)} \quad \cdots \quad \frac{\partial u_h(t)}{\partial W_j^{1,0}(t)} \quad \frac{\partial F_1^{1,0}(t-1)}{\partial W_j^{1,0}(t)} \quad \cdots \quad \frac{\partial F_m^{1,0}(t-1)}{\partial W_j^{1,0}(t)}\right]^T\\ &\approx \left[0 \quad \cdots \quad 0 \quad \frac{\partial F_1^{1,0}(t-1)}{\partial W_j^{1,0}(t-1)} \quad \cdots \quad \frac{\partial F_m^{1,0}(t-1)}{\partial W_j^{1,0}(t-1)}\right]^T.\end{aligned} \quad (27)$$

performance in the cases that the weight norm is monotonically increasing (see footnote 1 and simulations). This is similar, to a certain extent, to the work of Hammer and Tino [9].

1) Summary of the eEBP Algorithm: The proposed eEBP iteration training algorithm for ENs can be summarized as follows.

- Step 0: Set a maximum training step as the stopping criterion (maybe an infinite of steps if a real-time system is required).
- Step 1: Form the new input vector $x(t)$.
- Step 2: Calculate the neural network output $y^{(2)}(t)$ and error $e(t)$ with the input and current estimated weights.
- Step 3: Update the weights and adaptive dead zone of each layer through (15), (16), (19), and (20).
- Step 4: If the stopping criterion is not reached, return to Step 1 to continue the iteration.

III. Weight Convergence Proof of the Nonlinear eEBP Training

As discussed in the introduction, standard gradient-descent training methods for ENs can face severe problems [7], [8] and the design of efficient training algorithms for ENs and recurrent networks remains a challenging problem. In particular, this section shows that the proposed new eEBP training will guarantee weight convergence, which can prevent weight drift/divergence and in turn lead to a relatively smaller weight norm to improve the generalization performance of ENs [9].

A. Convergence Proof of the Output Layer

To perform a rigorous convergence proof of the weight norm, which is different from the linear output-layer case [13], [14], an optimal or suboptimal prediction error $e^{2,1}(t)$ of the output layer is first defined, based on the optimal or suboptimal output-layer weight $W^{*2,1}$

$$\begin{aligned} e^{2,1}(t) &= F(W^{2,1}(t)F^{1,0}(t)) - F(W^{*2,1}F^{1,0}(t)) \\ &= \mu^{2,1}(t)\tilde{W}^{2,1}(t)F^{1,0}(t) \end{aligned} \tag{29}$$

a temporary variable to establish the weight convergence result of the nonlinear recursive eEBP learning algorithm for the output layer. It is derived according to mean value theory with the positive variable matrix $\mu^{2,1}(t) = \mathrm{diag}[\mu_1^{2,1}(t) \quad \cdots \quad \mu_h^{2,1}(t)]$ and $0 < \mu_k^{2,1}(t) \le 1\ (1 \le k \le h)$ (similar to footnote 2, the maximum of the mean value is smaller than or equal to 1). Here $W^{*2,1} \in R^{h\times m}$ and $\tilde{W}^{2,1}(t) = (W^{2,1}(t) - W^{*2,1}) = \{\tilde{W}_{kj}^{2,1}(t)\}_{h\times m} \in R^{h\times m}$ $(1 \le k \le h, 1 \le j \le m)$ are the optimal or suboptimal weight and weight error matrices of the output layer, which implies that the gradient algorithm will achieve a local minimum rather than a global minimum for the eEBP objective function (10).

As for the definition of $F^{1,0}(t)$, the optimal or suboptimal output of hidden layer can also be defined as (30) shown at the bottom of the page, where $x^*(t) \in R^{n\times 1}$ is the optimal or suboptimal input state and $W^{*2,1} \in R^{m\times n}$ is the optimal or suboptimal weight of the hidden layer of the ENs.

The prediction error of the ENs can then be extended as

$$e(t) = u(t) - y^{(2)}(t) + \nu(t) \tag{31}$$

$$= F(W^{*2,1}F^{*1,0}(t)) - F(W^{2,1}(t)F^{1,0}(t)) + \nu(t) \tag{32}$$

$$\begin{aligned} &= [F(W^{*2,1}F^{*1,0}(t)) - F(W^{*2,1}F^{1,0}(t))] \\ &\quad - [F(W^{2,1}(t)F^{1,0}(t)) - F(W^{*2,1}F^{1,0}(t))] + \nu(t) \end{aligned} \tag{33}$$

$$= -e^{2,1}(t) + \tilde{\nu}^{2,1}(t) \tag{34}$$

where the bounded equivalent disturbance

$$\tilde{\nu}^{2,1}(t) = \nu(t) + F(W^{*2,1}F^{*1,0}(t)) - F(W^{*2,1}F^{1,0}(t)) \tag{35}$$

is a combined error produced from the prediction uncertainty $\nu(t)$ and the further hidden-layer training error $F(W^{*2,1}F^{*1,0}(t)) - F(W^{*2,1}F^{1,0}(t))$. Then, the eEBP training error can be rewritten as

$$e(t) = \tilde{\nu}^{2,1}(t) - e^{2,1}(t). \tag{36}$$

Multiplying both sides of (36) by $e^T(t)$ yields

$$e^T(t)e^{2,1}(t) = e^T(t)\tilde{\nu}^{2,1}(t) - e^T(t)e(t). \tag{37}$$

Because $e(t) = [e_1(t) \dots e_h(t)]^T$, $e^{2,1}(t) = [e_1^{2,1}(t) \dots e_h^{2,1}(t)]^T$, and $\tilde{\nu}^{2,1}(t) = [\tilde{\nu}_1^{2,1}(t) \quad \cdots \quad \tilde{\nu}_h^{2,1}(t)]^T$ are all h-dimensional vectors, (37) can be rewritten as

$$\sum_{k=1}^{h} e_k(t)e_k^{2,1}(t) = \sum_{k=1}^{h} e_k(t)\tilde{\nu}_k^{2,1}(t) - \sum_{k=1}^{h} e_k^2(t) \tag{38}$$

$$\Rightarrow e_k(t)e_k^{2,1}(t) = e_k(t)\tilde{\nu}_k^{2,1}(t) - e_k^2(t). \tag{39}$$

Note that (37)–(39) establish an important relation between the ENs' prediction error $e(t)$ and the optimal or suboptimal estimation error $e^{2,1}(t)$ of the output layer, which can be used in the following theorem.

Theorem 1: Assume that there exists an optimal or suboptimal weight matrix $W^{*2,1}$ and an unknown optimal or suboptimal dead zone vector

$$\Delta^{*2,1} = [\,\Delta_1^{*2,1} \quad \cdots \quad \Delta_h^{*2,1}\,]^T \tag{40}$$

where $\Delta_k^{*2,1} \ge (F_k'^{2,1}(t))/(\mu_k^{2,1}(t))|\tilde{\nu}_k^{2,1}(t)|$, $1 \le k \le h$, are unknown up-bounds of the bounded equivalent disturbance related terms (see Remark 2 for more details). If the output layer of the ENs is trained by the adaptive training algorithm (15), the

$$F^{*1,0}(t) = [\,f(W_1^{*1,0}(t)x^*(t)) \quad f(W_2^{*1,0}(t)x^*(t)) \quad \cdots \quad f(W_m^{*1,0}x^*(t))\,]^T \tag{30}$$

weight matrix $W^{2,1}(t)$ and the dead zone estimate $\Delta^{2,1}(t)$ are guaranteed to be convergent in the Lyapunov sense

$$\|\tilde{W}^{2,1}(t+1)\|^2 - \|\tilde{W}^{2,1}(t)\|^2 + \|\tilde{\Delta}^{2,1}(t+1)\|^2 - \|\tilde{\Delta}^{2,1}(t)\|^2 \le 0 \qquad \forall t$$

with $\tilde{W}^{2,1}(t) = W^{2,1}(t) - W^{*2,1}$ and $\tilde{\Delta}^{2,1}(t) = \Delta^{2,1}(t) - \Delta^{*2,1}$.

Furthermore, the estimate error of the ENs is bounded as

$$\lim_{t \to \infty} \sup \|e(t)\| \le \|\Delta^{*2,1}\|. \tag{41}$$

Proof: The optimal or suboptimal weight matrix $W^{*2,1}$ and adaptive dead zone vector $\Delta^{*2,1}$ are subtracted from both sides of (15) and (16), respectively, squared, and the Lyapunov function constructed via (39) to obtain

$$\begin{aligned}
&\left\|\tilde{W}^{2,1}(t+1)\right\|^2 - \left\|\tilde{W}^{2,1}(t)\right\|^2 + \left\|\tilde{\Delta}^{2,1}(t+1)\right\|^2 - \left\|\tilde{\Delta}^{2,1}(t)\right\|^2 \\
&= \left[2\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} e^T(t) F'^{2,1}(t) \tilde{W}^{2,1}(t) F^{1,0}(t)\right] \\
&\quad + \left\|\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} F^{1,0}(t) e^T(t) F'^{2,1}(t)\right\|^2 \\
&\quad + 2\bar{e}^T(t)\tilde{\Delta}^{2,1}(t)\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} + \left\|\bar{e}(t)\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)}\right\|^2 \\
&= \left[2\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} e^T(t) F'^{2,1}(t) [\mu^{2,1}(t)]^{-1} [\mu^{2,1}(t)] \tilde{W}^{2,1}(t) F^{1,0}(t)\right] \\
&\quad + \left\|\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} F^{1,0}(t) e^T(t) F'^{2,1}(t)\right\|^2 \\
&\quad + 2\bar{e}^T(t)\tilde{\Delta}^{2,1}(t)\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} + \left\|\bar{e}(t)\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)}\right\|^2 \\
&= \sum_{k=1}^{h} 2\frac{F_k'^{2,1}(t)}{\mu_k^{2,1}(t)} e_k(t) e_k^{2,1}(t)\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} \\
&\quad + \left\|\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} F^{1,0}(t) e^T(t) F'^{2,1}(t)\right\|^2 \\
&\quad + 2\bar{e}^T(t)\tilde{\Delta}^{2,1}(t)\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} + \left\|\bar{e}(t)\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)}\right\|^2 \\
&= \sum_{k=1}^{h} 2\frac{F_k'^{2,1}(t)}{\mu_k^{2,1}(t)} (e_k(t)\tilde{\nu}_k^{2,1}(t) - e_k^2(t))\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} \\
&\quad + \left\|\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} F^{1,0}(t) e^T(t) F'^{2,1}(t)\right\|^2 \\
&\quad + 2\bar{e}^T(t)\left(\Delta^{2,1}(t) - \Delta^{*2,1}\right)\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} + \left\|\bar{e}(t)\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)}\right\|^2 \\
&\le \sum_{k=1}^{h} 2\left[|e_k(t)|\left(\frac{F_k'^{2,1}(t)}{\mu_k^{2,1}(t)}|\tilde{\nu}_k^{2,1}(t)| - \Delta_k^{*2,1}\right) - \frac{F_k'^{2,1}(t)}{\mu_k^{2,1}(t)} e_k^2(t)\right] \\
&\quad \times \frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} + \left\|\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} F^{1,0}(t) e^T(t) F'^{2,1}(t)\right\|^2 \\
&\quad + 2\bar{e}^T(t)\left(\Delta^{2,1}(t)\right)\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} + \left\|\bar{e}(t)\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)}\right\|^2 \\
&\le \sum_{k=1}^{h} -2\frac{F_k'^{2,1}(t)}{\mu_k^{2,1}(t)} e_k^2(t)\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} \\
&\quad + \left\|\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} F^{1,0}(t) e^T(t) F'^{2,1}(t)\right\|^2 \\
&\quad + 2\bar{e}^T(t)\Delta^{2,1}(t)\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} + \left\|\bar{e}(t)\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)}\right\|^2.
\end{aligned} \tag{42}$$

The last equality of (42) is derived from (39) and the definition of $\tilde{\Delta}^{2,1}(t)$. The last inequality of (42) is based on definitions of the optimal or suboptimal adaptive dead zone components $\Delta_k^{*2,1} \ge (F_k'^{2,1}(t))/(\mu_k^{2,1}(t))|\tilde{\nu}_k^{2,1}(t)|$, $1 \le k \le h$, as in (40) [note $(F_k'^{2,1}(t))/(\mu_k^{2,1}(t)) \ge 0$]. Then, we can rewrite (42) as [note $\|\bar{e}(t)\| = \|e(t)\|$]

$$\begin{aligned}
&\left\|\tilde{W}^{2,1}(t+1)\right\|^2 - \left\|\tilde{W}^{2,1}(t)\right\|^2 + \left\|\tilde{\Delta}^{2,1}(t+1)\right\|^2 - \left\|\tilde{\Delta}^{2,1}(t)\right\|^2 \\
&\le \sum_{k=1}^{h} -2\frac{F_k'^{2,1}(t)}{\mu_k^{2,1}(t)} e_k^2(t)\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} \\
&\quad + \left\|\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} F^{1,0}(t) e^T(t) F'^{2,1}(t)\right\|^2 \\
&\quad + 2\bar{e}^T(t)\Delta^{2,1}(t)\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} + \left\|\bar{e}(t)\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)}\right\|^2 \\
&\le -F_{\min}'^{2,1}\|e(t)\|^2 \frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} + 2\left\|e^T(t)\right\|\left\|\Delta^{2,1}(t)\right\|\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)} \\
&\quad + \left(\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)}\right)^2 \|e(t)\|^2 \left(1 + \left\|F^{1,0}(t)\right\|^2 \left\|F'^{2,1}(t)\right\|^2\right) \\
&\le -\frac{\epsilon^{2,1}(t)}{\rho^{2,1}(t)}\|e(t)\|\left(\|e(t)\| - \left\|\Delta^{2,1}(t)\right\|\right) \le 0.
\end{aligned} \tag{43}$$

Note that the second to last inequality is derived from the normalization factor (18) and the last inequality is from the definition of the adaptive learning rate (17).

Furthermore, the Lyapunov function

$$\left\|\tilde{W}^{2,1}(t+1)\right\|^2 - \left\|\tilde{W}^{2,1}(t)\right\|^2 + \left\|\tilde{\Delta}^{2,1}(t+1)\right\|^2 - \left\|\tilde{\Delta}^{2,1}(t)\right\|^2$$

in (43) is bounded below by zero and is not increasing; the limit therefore exists and the last item of (43) will go to zero, that is

$$\lim_{t \to \infty} \sup \|e(t)\| \le \|\Delta^{*2,1}\| \tag{44}$$

which completes the proof.

Remark 2:

- One of the key contributions of this paper is to use the multivariable adaptive dead zone, which is new and not proposed by previous researches. The critical role of the multivariable dead zone concept is shown in the proof of Theorem 1. For the second last inequality of (42), the items in the round bracket can be eventually deleted by enlarging the equation, because the adaptive dead zone components are estimated separately (even though only the norm of dead zone is used for the update of adaptive learning rate). Therefore, a scalar dead zone parameter is not enough to

handle weight convergence of the MIMO system, at least, in the theoretical sense.

- The last inequality of (43) shows the crucial role of the adaptive learning rate $\epsilon^{2,1}(t)$ as defined in (17), which cuts off the training whenever the training error is too small, that is, $\|e(t)\| \leq \left\|\Delta^{2,1}(t)\right\|$, and guarantees that the weight error norm vector $\tilde{W}^{2,1}(t)$ of the output layer is nonincreasing for every discrete time step t and converges to an optimal or suboptimal value in the Lyapunov sense to improve the generalization performance (see the simulation in Section IV for more details). In general, the weight convergence condition (43) cannot be guaranteed for a fixed learning rate ϵ of EBP learning, as in (11), if the maximum norm of the equivalent disturbance $\tilde{\nu}^{2,1}(t)$, representing the prediction error with system uncertainty, is not zero and can cause weight drift or divergence [30]. Thus, linear prediction filtering theory has been extended to the nonlinear neuron case and is indeed suitable for the one-step-ahead discrete-time sequence prediction problem.
- The adaptive dead zone vector $\Delta^{2,1}(t)$ can be estimated online for a MIMO system. To the best of our knowledge, this is the first time the adaptive dead zone concept has been applied to a MIMO system with proof of guaranteed convergence, which is a crucial difference between this paper and previous approaches in [13], [14], and [30].
- The assumption of the optimal or suboptimal dead zone estimate $\Delta_k^{*2,1}(t)$ being up-bounds of $(F_k'^{2,1}(t))/(\mu_k^{2,1}(t))|\tilde{\nu}_k^{2,1}(t)|$, $1 \leq k \leq h$, is reasonable, since $(F_k'^{2,1}(t))/(\mu_k^{2,1}(t))|\tilde{\nu}_k^{2,1}(t)|$ are the bounded variables for output-layer training. The optimal or suboptimal weight matrices $W^{*2,1}$ and $W^{*1,0}$, and related variables such as $F^{*1,0}(t)$ and the equivalent disturbance $\tilde{\nu}^{2,1}(t)$ are also bounded temporary variables and only useful for the convergence proof and are not used in the eEBP training algorithm of Section II.

B. Convergence Proof of the Hidden Layer

Similar to the output-layer case (31), the estimate error of the ENs can be extended as

$$\begin{aligned} e(t) &= u(t) - y^{(0)}(t) + \nu(t) \\ &= F(W^{*2,1}F^{*1,0}(t)) - F(W^{2,1}(t)F^{1,0}(t)) + \nu(t) \\ &= F(W^{*2,1}F^{*1,0}(t)) - F(W^{2,1}(t)F(W^{*1,0}x(t))) \\ &\quad + F(W^{2,1}(t)F(W^{*1,0}x(t))) \\ &\quad - F(W^{2,1}(t)F(W^{1,0}(t)x(t))) + \nu(t) \\ &= F(W^{2,1}(t)F(W^{*1,0}x(t))) \\ &\quad - F(W^{2,1}(t)F(W^{1,0}(t)x(t))) + \nu^{1,0}(t) \\ &= z - e^{1,0}(t) + \tilde{\nu}^{1,0}(t) \end{aligned} \tag{45}$$

where

$$\begin{aligned} \tilde{\nu}^{1,0}(t) &= F(W^{*2,1}F^{*1,0}(t)) - F(W^{2,1}(t)F(W^{*1,0}x(t))) + \nu(t) \\ &= \sum_{j=1}^{m} \tilde{\nu}_j^{1,0}(t) \end{aligned} \tag{46}$$

with $\tilde{\nu}_j^{1,0}(t) \in R^{h\times 1}$.

The optimal or suboptimal training error of the hidden layer $e^{1,0}(t)$ can be extended as

$$\begin{aligned} e^{1,0}(t) &= F(W^{2,1}(t)F(W^{1,0}(t)x(t))) \\ &\quad - F(W^{2,1}(t)F(W^{*1,0}x(t))) \\ &= \mu_1(t)W_1^{2,1}(t)\tilde{W}_1^{1,0}(t)x(t) \\ &\quad + \mu_2(t)W_2^{2,1}(t)\tilde{W}_2^{1,0}(t)x(t) + \cdots \\ &\quad + \mu_m(t)W_m^{2,1}(t)\tilde{W}_m^{1,0}(t)x(t) \\ &= \sum_{j=1}^{m} \mu_j(t)W_j^{2,1}(t)\tilde{W}_j^{1,0}(t)x(t) \\ &= \sum_{j=1}^{m} e_j^{1,0}(t) \end{aligned} \tag{47}$$

where $e_j^{1,0}(t) = \mu_j(t)W_j^{2,1}(t)\tilde{W}_j^{1,0}(t)x(t)$ with $\tilde{W}_j^{1,0}(t) = W_j^{1,0}(t) - W_j^{*1,0} \in R^{1\times n}$, which is the vector difference between the jth row of the estimated $W^{1,0}(t)$ and the optimal or suboptimal weight $W^{*1,0}$, $W_j^{2,1}(t)$ is the jth column of the estimated $W^{2,1}(t)$, and $\mu_j(t) = \mathrm{diag}[\mu_{1j}(t)\ldots\mu_{hj}(t)]$, with $0 \leq \mu_{kj}(t) \leq 1$, $\forall k, j$, being the mean value of the nonlinear activation functions.

According to (45) and similar to the output-layer case, it is assumed that

$$e_j^{1,0}(t) = \frac{-e(t)}{m} + \tilde{\nu}_j^{1,0}(t). \tag{48}$$

Theorem 2: The weight matrix $W^{1,0}(t)$ and the adaptive dead zone parameter $\Delta^{1,0}(t)$ are guaranteed to be convergent in the Lyapunov sense

$$\left\|\tilde{W}^{1,0}(t+1)\right\|^2 - \left\|\tilde{W}^{1,0}(t)\right\|^2 + \left(\tilde{\Delta}^{1,0}(t+1)\right)^2 - \left(\tilde{\Delta}^{1,0}(t)\right)^2 \leq 0$$

with $\tilde{W}^{1,0}(t) = W^{1,0}(t) - W^{*1,0}$, $\tilde{\Delta}^{1,0}(t) = \Delta^{1,0}(t) - \Delta^{*1,0}$, where $W^{*1,0}$ represents an optimal or suboptimal weight matrix for the minimum point of the extended gradient, $\Delta^{*1,0} \geq m(\tilde{\nu}_{\max}^{1,0}/\mu_{\min})(1 + \gamma^{1,0}(t)\Delta_W^{1,0}(t))$ represents an unknown optimal or suboptimal dead zone value with $\tilde{\nu}_{\max}^{1,0} = \max(|\tilde{\nu}^{1,0}(t)|)$, $\forall t$, and $\mu_{\min} = \min(\|\mu_j(t)\|) \neq 0$, $\forall j$, t. Furthermore, similar to Theorem 1

$$\lim_{t\to\infty} \sup \|e(t)\| \leq \Delta^{*1,0}. \tag{49}$$

Proof: We use the trace property

$$\begin{aligned} &-\mathrm{tr}\left\{\left[\frac{\partial E(t)}{\partial W^{1,0}(t)} + \gamma^{1,0}(t)\frac{\partial E(t)}{\partial x(t)}\frac{\partial x(t)}{\partial W^{1,0}(t)}\right]\tilde{W}^{1,0T}(t)\right\} \\ &= \mathrm{tr}\left\{F'^{1,0}(t)W^{2,1T}(t)F'^{2,1}(t)e(t)x^T(t)\tilde{W}^{1,0T}(t)\right\} \\ &\quad + \gamma^{1,0}(t)\,\mathrm{tr}\left\{F'^{1,0}(t)W^{2,1T}(t)F'^{2,1}(t)e(t)\underline{W}^{1,0}(t)\right. \\ &\qquad \left.\times \frac{\partial x(t)}{\partial W^{1,0}(t)}\tilde{W}^{1,0T}(t)\right\} \\ &= \mathrm{tr}\left\{F'^{1,0}(t)W^{2,1T}(t)F'^{2,1}(t)e(t)x^T(t)\tilde{W}^{1,0T}(t)\right\} \end{aligned}$$

$$
\begin{aligned}
&+\gamma^{1,0}(t)\,\mathrm{tr}\Bigg\{F'^{1,0}(t)W^{2,1T}(t)F'^{2,1}(t)e(t)\underline{W}^{1,0}(t)\\
&\qquad\times\frac{\partial x(t)}{\partial W^{1,0}(t)}\{x(t)x^T(t)\}^{-1}x(t)\Bigg\}\\
&=x^T(t)\tilde{W}^{1,0T}(t)F'^{1,0}(t)W^{2,1T}(t)F'^{2,1}(t)e(t)\\
&\quad+\gamma^{1,0}(t)x^T(t)\tilde{W}^{1,0T}(t)F'^{1,0}(t)W^{2,1T}(t)F'^{2,1}(t)\\
&\quad\times e(t)\underline{W}^{1,0}(t)\frac{\partial x(t)}{\partial W^{1,0}(t)}\bar{x}(t)x(t)\\
&=\sum_{j=1}^{m}\Big[F_j'^{1,0}(t)e^T(t)F'^{2,1}(t)(\mu_j(t))^{-1}\mu_j(t)W_j^{2,1}(t)\\
&\qquad\times\tilde{W}_j^{1,0}(t)x(t)\Big]\left(1+\gamma^{1,0}(t)\Delta_W^{1,0}(t)\right)\\
&=\sum_{j=1}^{m}\left[F_j'^{1,0}(t)e^T(t)F'^{2,1}(t)(\mu_j(t))^{-1}e_j^{1,0}(t)\right]\\
&\quad\times\left(1+\gamma^{1,0}(t)\Delta_W^{1,0}(t)\right)\\
&=\sum_{j=1}^{m}\Bigg[\frac{-F_j'^{1,0}(t)e^T(t)F'^{2,1}(t)(\mu_j(t))^{-1}e(t)}{m}\\
&\qquad+F_j'^{1,0}(t)\tilde{\nu}_j^{1,0T}(t)F'^{2,1}(t)(\mu_j(t))^{-1}e(t)\Bigg]\\
&\quad\times\left(1+\gamma^{1,0}(t)\Delta_W^{1,0}(t)\right)\\
&\le-F_{\min}'^{1,0}F_{\min}'^{2,1}\|e(t)\|^2\\
&\quad+m\frac{\tilde{\nu}_{\max}^{1,0}}{\mu_{\min}}\left(1+\gamma^{1,0}(t)\Delta_W^{1,0}(t)\right)\|e(t)\| \qquad (50)
\end{aligned}
$$

with $F_{\min}'^{1,0}=\min(F_j'^{1,0}(t))\neq 0,\ \forall t,\ j$.

The following derivation is similar to the output-layer case in Section III-A. Subtracting $W^{*1,0}$ and $\Delta^{*1,0}$ and then squaring both sides of the recursive (19) and (20), using (50), (23), (22), (21), and (28) yields a similar derivation based on (50)

$$
\begin{aligned}
&\left\|\tilde{W}^{1,0}(t+1)\right\|^2-\left\|\tilde{W}^{1,0}(t)\right\|^2+\left(\tilde{\Delta}^{1,0}(t+1)\right)^2-\left(\tilde{\Delta}^{1,0}(t)\right)^2\\
&=-2\,\mathrm{tr}\Bigg\{\frac{\epsilon^{1,0}(t)}{\rho^{1,0}(t)}\Bigg[\frac{\partial E(t)}{\partial W^{1,0}(t)}\\
&\qquad+\gamma^{1,0}\frac{\partial E(t)}{\partial x(t)}\frac{\partial x(t)}{\partial W^{1,0}(t)}\Bigg]\tilde{W}^{1,0T}(t)\Bigg\}\\
&\quad+\left\|\frac{\epsilon^{1,0}(t)}{\rho^{1,0}(t)}\left[\frac{\partial E(t)}{\partial W^{1,0}(t)}+\gamma^{1,0}(t)\frac{\partial E(t)}{\partial x(t)}\frac{\partial x(t)}{\partial W^{1,0}(t)}\right]\right\|^2\\
&\quad+2\|e(t)\|\tilde{\Delta}^{1,0}(t)\frac{\epsilon^{1,0}(t)}{\rho^{1,0}(t)}+\|e(t)\|^2\left(\frac{\epsilon^{1,0}(t)}{\rho^{1,0}(t)}\right)^2\\
&\le 2\frac{\epsilon^{1,0}(t)}{\rho^{1,0}(t)}\Bigg[-F_{\min}'^{1,0}F_{\min}'^{2,1}\|e(t)\|^2\\
&\qquad+2m\frac{\tilde{\nu}_{\max}^{1,0}}{\mu_{\min}}\left(1+\gamma^{1,0}(t)\Delta_W^{1,0}(t)\right)\|e(t)\|\Bigg]\\
&\quad+\left(\frac{\epsilon^{1,0}(t)}{\rho^{1,0}(t)}\right)^2\|e(t)\|^2\left\|F'^{1,0}(t)W^{2,1T}(t)F'^{2,1}(t)\right\|^2\\
&\quad\times\left\|x^T(t)+\gamma^{1,0}(t)\underline{W}^{1,0}(t)\right\|^2\\
&\quad+2\|e(t)\|\tilde{\Delta}^{1,0}(t)\frac{\epsilon^{1,0}(t)}{\rho^{1,0}(t)}+\|e(t)\|^2\left(\frac{\epsilon^{1,0}(t)}{\rho^{1,0}(t)}\right)^2\\
&\le-2F_{\min}'^{1,0}F_{\min}'^{2,1}\|e(t)\|\frac{\epsilon^{1,0}(t)}{\rho^{1,0}(t)}\left[\|e(t)\|-\Delta^{1,0}(t)\right]\le 0. \qquad (51)
\end{aligned}
$$

Note that definition (22) is used for the proof and the role of $\gamma^{1,0}(t)$ is to guarantee that $1+\gamma^{1,0}(t)\Delta_W^{1,0}(t)\ge 0$ is a bounded positive variable, which completes the proof.

Remark 3:

- Similar to previous research [13], to some extent weight convergence analysis provides certain clues for improving the training performance of the ENs. For example, the optimal or suboptimal adaptive dead zone value $\Delta^{*1,0}\ge m(\tilde{\nu}_{\max}^{1,0}/\mu_{\min})\left(1+\gamma^{1,0}(t)\Delta_W^{1,0}(t)\right)$ indicates that too many hidden layers should not be selected for an unnecessarily larger network structure, because use of a bigger m would be implied, which would result in a larger online training error.
- The extended gradient $\Delta_W^{1,0}(t)$ is cut off whenever it becomes unbounded or negative according the definition of $\gamma^{1,0}(t)$ in (22).

IV. Simulations

Example 1 (XOR): The exclusive or XOR function [1] is presented as a problem involving two-bit input vectors (00, 11, 01, 10) yielding one-bit output vectors (0, 0, 1, 1, respectively). The inputs are concatenated and presented as an unbroken sequence. Elman's version of the XOR problem is followed and the input consisted of a sequence of 10 000 bits constructed in this manner. This input stream is presented to the ENs (with one input unit, two hidden units, one output unit, and two context units), one bit at a time. The task of the network is, at each point in time, to predict the next bit in the sequence, that is, one-step-ahead prediction, as mentioned in Section II. In other words, given the input sequence shown, where one bit at a time is presented, the correct output at corresponding points in time is

Input: 101000011110101 ...

Output: 01000011110101? ...

Fig. 2(a)–(c) shows that standard EBP training using a high learning rate produces a larger training error, while lower learning rates can result in slow learning procedures. In particular, EBP training for ENs can lead to poor generalization performance. For example, as shown in Fig. 3(c), the low learning rate $\epsilon=0.01$ reduces the training error to a very low level in the first 5000 steps. The error increases again, however, after 5000 steps because of the overfitting problem [9], [13], [29], which leads to poor generalization performance. In contrast, the new eEBP training, as shown in Fig. 2(d), provides an optimal or suboptimal tradeoff between the learning speed (within fewer than 50 discrete time steps) of the inherent sequence dynamics and good generalization performance (for the remaining 9950 steps). This is because eEBP uses the extended gradient to capture the extra information of the time structure

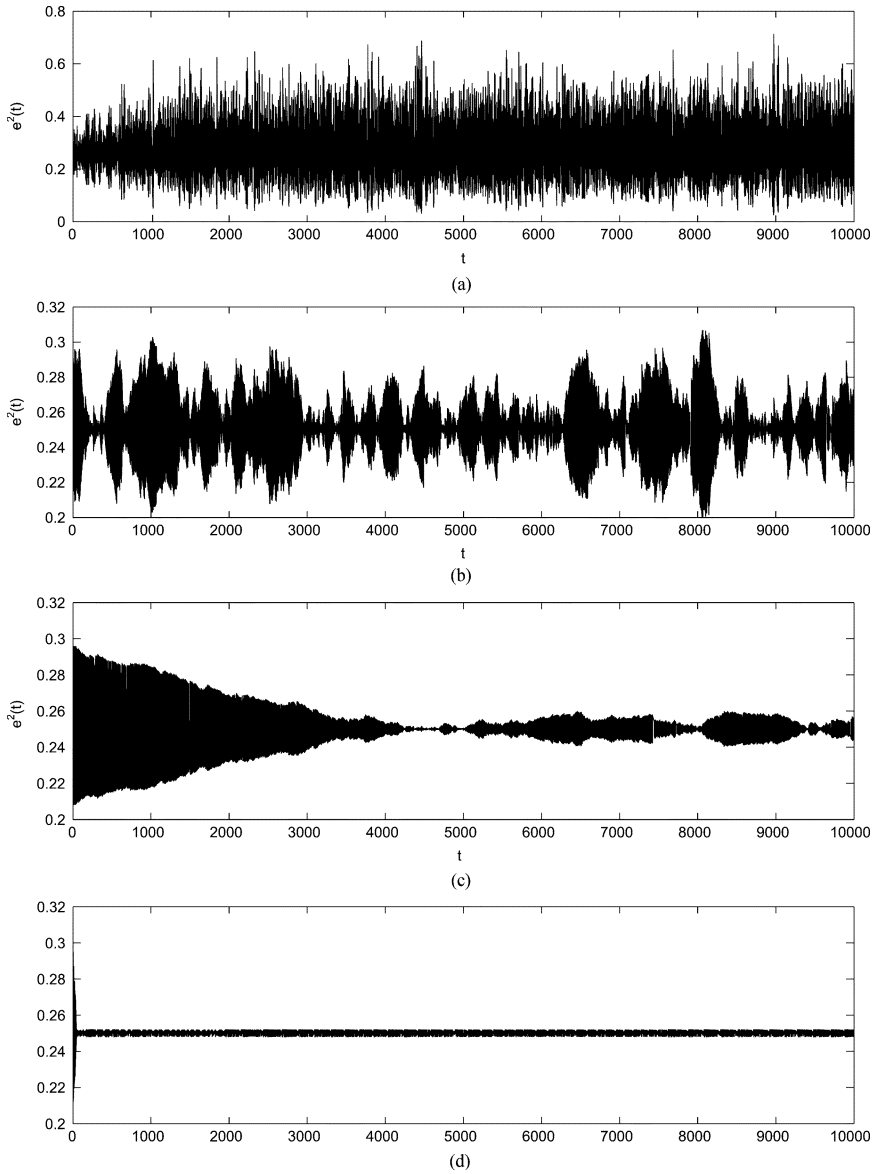

Fig. 2. Tracking of the prediction error norms $\|e(t)\|$. (a) EBP training with learning rate $\epsilon = 1$. (b) EBP training with learning rate $\epsilon = 0.1$. (c) EBP training with learning rate $\epsilon = 0.01$. (d) eEBP training.

and automatically switches the training procedure on/off, according to the adaptive learning rates. Fig. 3 shows that the hidden-layer weight norms of the eEBP converge to at least a local optimal or suboptimal value. In contrast, standard EBP training based on a high learning rate produces poor learning performance with the largest hidden-layer weight norm, which may not be desirable [5]. An EBP using a smaller learning rate leads to not only a slow learning speed but also a relatively poor generalization performance compared to eEBP, even though it may also lead to relatively smaller weight norms here. The argument is that better generalization requires not only a small weight norm but also an optimal or suboptimal weight value in the weight monotonically increasing examples.

The details of the XOR problem can be further analyzed from the initial period of eEBP training. Fig. 4 shows the first 100 steps of the hidden-layer and output-layer responses together with the adaptive learning rates. The results indicate that most of the neural learning activities switch on near the high-level output points of the neurons to automatically gain maximum information during the eEBP training procedure. In fact, as shown in Fig. 4(d), the error signal of the eEBP training was reduced to the minimum level around $(\|e(t)\| = 0.5)$ after fewer than 50 steps. Assuming that the ENs' output will be converted to a truncated output for the values 1 and 0, the semirandom prediction will only be affected by the random effect but time structure of XOR. Therefore, the rest of the prediction error (after the first 50 steps) will remain at the same level for the eEBP to represent the generalization performance due to limited subregularities (only four different XOR combinations), compared to that of the standard EBP training, which needs much longer training times, as shown in Fig. 2.

Example 2: The XOR pattern [1] is simple because it involves single-bit inputs, with just four different input patterns, and the memory required needs to extend only one bit back in time. Multibit inputs of greater temporal extent with a higher number of possible sequences as well as patterns of variable duration,

TABLE I
VECTOR DEFINITIONS OF THE ALPHABET.

	Consonant	Vowel	Interrupted	High	Bock	Voiced
b	1	0	1	0	0	1
d	1	0	1	1	0	1
g	1	0	1	0	1	1
a	0	1	0	0	1	1
i	0	1	0	1	0	1
u	0	1	0	1	1	1

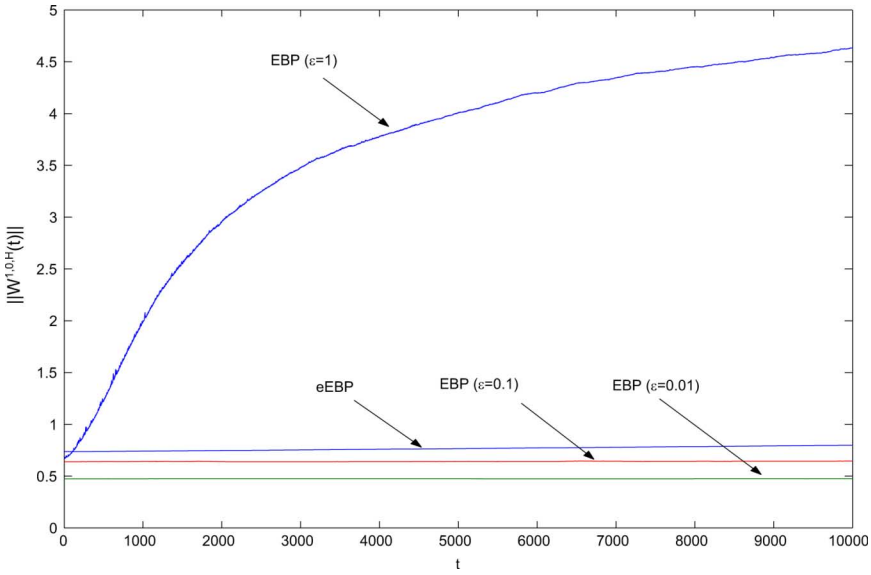

Fig. 3. Comparison of the weight convergence between EBP and eEBP training.

for example, would be more challenging. An input sequence providing these types of difficulties was therefore determined. Six different six-bit binary vectors comprised the sequence and, even though they were not derived from real speech, they can be thought of as speech sounds. The six dimensions of each vector would then correspond to characteristics of articulation. The vectors for the six letters are shown in Table I. The input sequence was devised in two stages: 1) the three consonants (b, d, g) were first randomly combined to obtain a sequence of 1000 letters, and 2) each consonant was then replaced according to the rules *b-bba*, *d-dii*, and *g-guuu*.

The sequence was semirandom, with consonants appearing randomly but always following a certain consonant and the identity and number of subsequent vowels were regular. The ENs were used to provide for the six-bit input vectors. The training regimen included six input units as well as 20 hidden units, six output units, and 20 context units. The six-bit input vectors were presented in sequence, one at a time, and the network was to predict the next input. The sequence was generated in 10 000 steps with the semirandom model so that the first pattern was presented after the last one. As in the XOR problem, the one-step prediction of the discrete-time sequence is solved again, that is, each of the input vectors $y^{(0)}(t)$ is one step ahead of the desired output $u(t)$, with $y^{(0)}(t+1) = u(t)$, to form the discrete-time prediction problem.

The ENs were trained by standard EBP via fixed learning rates through a semirandom sequence. It was then tested on another 20 semirandom sequences that obeyed the same regularities but that were created from different random initializations. The error signal for part of the training phase is shown in Fig. 5. The best EBP training result is shown in Fig. 5(b), with the fixed learning rate $\epsilon = 0.5$. The eEBP training result in Fig. 5(d) is clearly better than all the EBP training results. The basic reasons for this improvement are that the extended gradient was used to learn more of the inherent time structure of the sequence because of the extra information obtained from (24). Furthermore, the adaptive learning rate via the MIMO adaptive dead zone vector plays a crucial role. One can see that eEBP learns faster than EBP, in the sense that the training error is more rapidly reduced than for the EBP counterparts, as shown in Fig. 5. The adaptive learning rate controls the learning procedure by automatically switching learning rates $\epsilon^{1,0}(t)$ and $\epsilon^{2,1}(t)$ on/off, as shown in Fig. 12. In fact, eEBP turns the training procedure on and off frequently after 3000 training steps.

The advantages of using the extended gradient can also be demonstrated by the individual bit prediction capability of eEBP training, which is better in terms of the first three bits compared to the best EBP $(\epsilon = 0.5)$ training, as shown in Figs. 6–11, which were especially achieved by the individual estimates of the adaptive dead zone vector $\Delta^{1,0}(t)$ to treat the different noise levels of the individual bits.

The first bit corresponds to the consonant feature and the fourth bit corresponds to the high feature. While all the consonants have the same value for the consonant feature, they differ in their high feature. The network here has learned the particular vowels that follow each consonant, including their number, re-

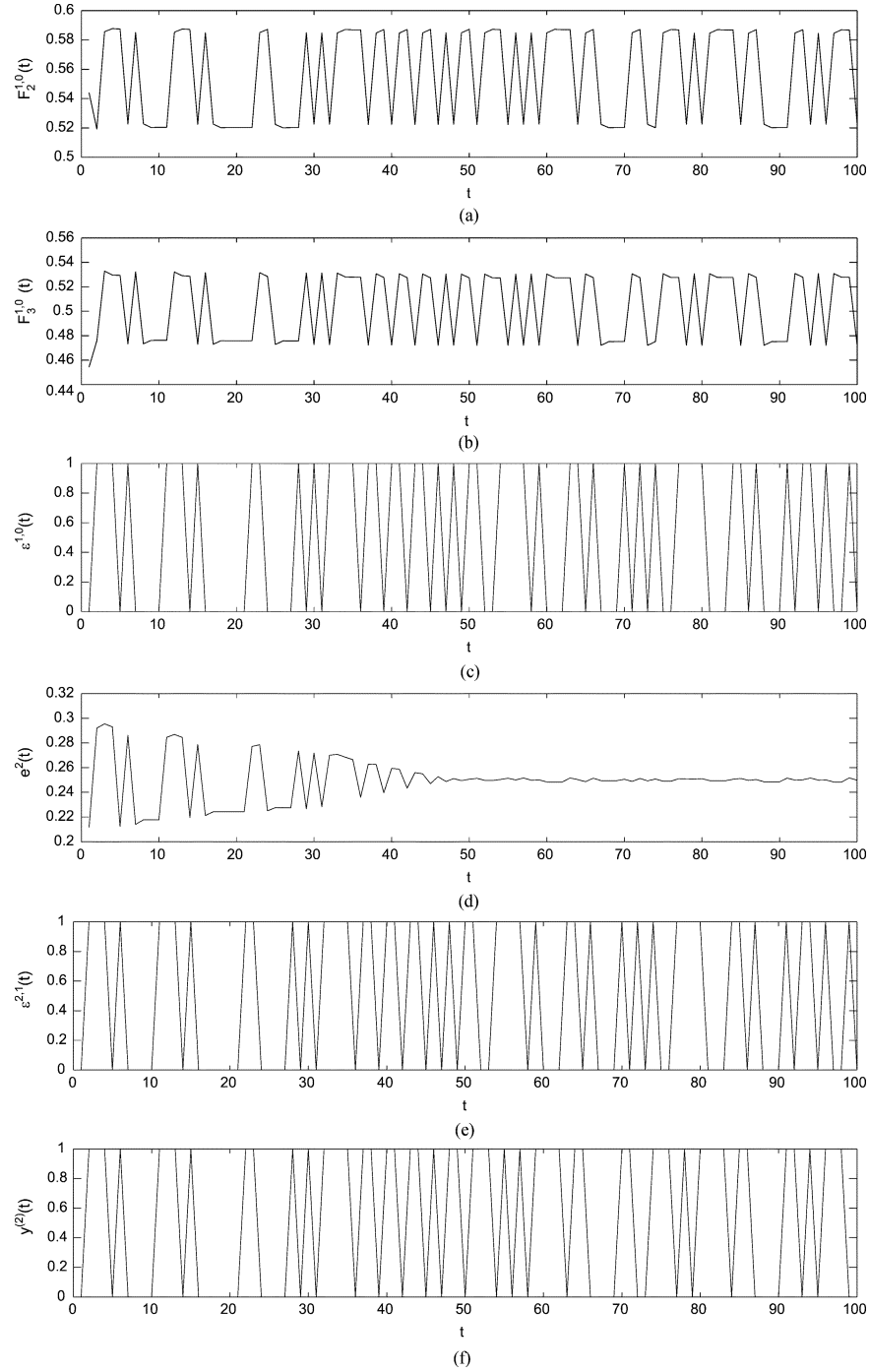

Fig. 4. Tracking of the hidden and output layers of the first 100 steps based on eEBP with the adaptive learning parameters for the XOR problem. (a) Tracking of the hidden-layer output $F_2^{1,0}(t)$. (b) Tracking of the hidden-layer output $F_3^{1,0}(t)$. (c) Tracking of the hidden-layer learning rate $\varepsilon^{1,0}(t)$ based on eEBP. (d) Tracking of the prediction square error $e^2(t)$ based on eEBP. (e) Tracking of the output-layer learning rate $\varepsilon^{2,1}(t)$ based on eEBP. (f) Tracking of the output $y^{(2)}(t)$ based on eEBP.

sulting in few errors on vowels. Interestingly, the network therefore also knows when to expect each consonant, even if it cannot successfully predict which one. This leads to few errors in predicting the bit patterns for consonant and frequent errors for the bit patterns for high and thus requires the use of context units (see Fig. 1). Simple FF network can learn the transitional probabilities from one input to another but not patterns spanning over two inputs. It is also interesting to note that some bit outputs are very close to each other (such as bit-one and bit-two of eEBP or bit-two and bit-three of EBP), which is similar to Example 1.

In terms of generalization capability, the simulation in Example 2 confirms Elman's [1] interesting point. In certain respects the input sequence is more complex than the XOR input. Even when complete predictions are not possible, subregularities at the level of individual bit patterns allow the network to make partial predictions. These partial predictions depend, however, on the structure of the input, which would imply that more extended sequential dependencies are not necessarily harder for networks to learn. Dependencies structured with the goal of an efficient training algorithm can therefore make learning easier.

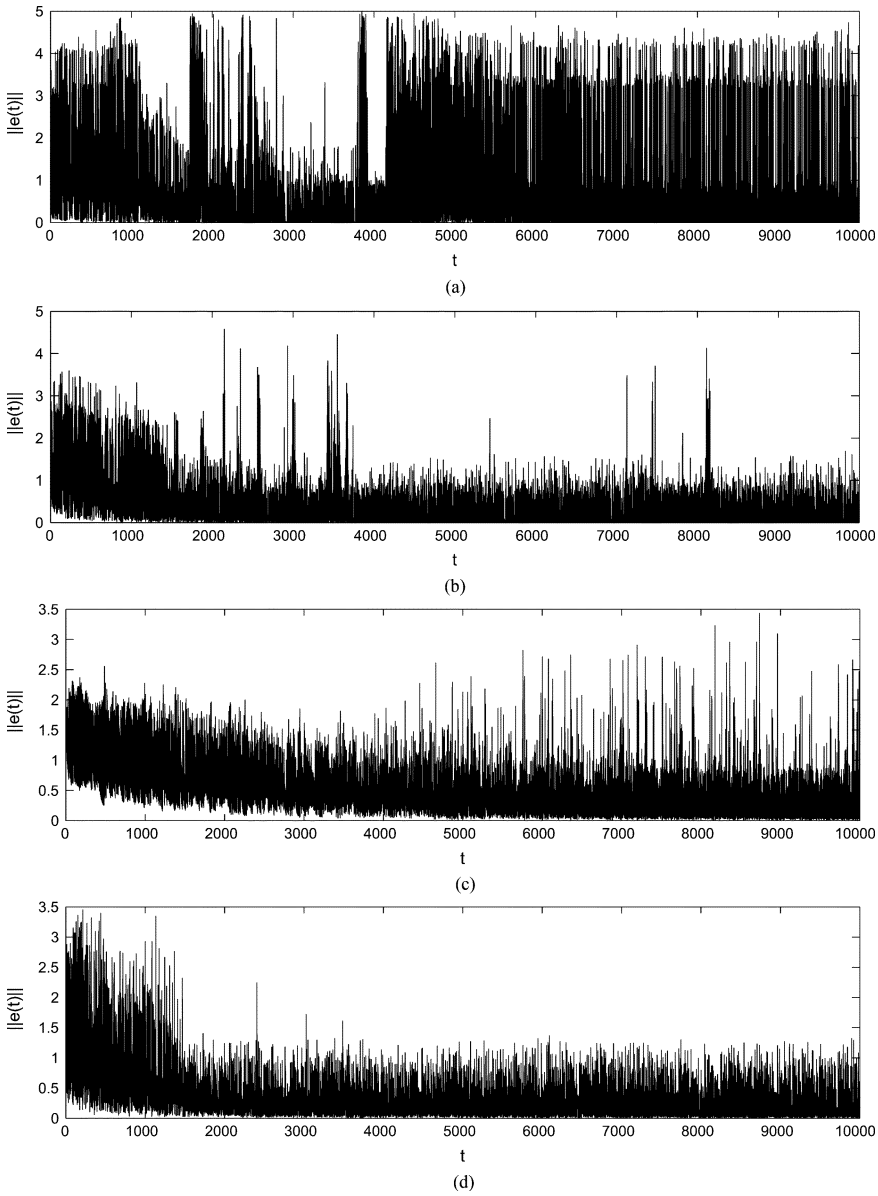

Fig. 5. Tracking of the prediction error norms $||e(t)||$ of Example 2. (a) EBP training with learning rate $\epsilon = 1$. (b) EBP training with learning rate $\epsilon = 0.5$. (c) EBP training with learning rate $\epsilon = 0.1$. (d) eEBP training.

The extended gradient and adaptive training rates of eEBP ensure that the weights converge at optimal or suboptimal values. The optimal or suboptimal fixed-weight matrices on which the test runs were based were obtained from the best EBP $(\epsilon = 0.5)$ and eEBP training, respectively. The sum (divided by 10 000 steps) of the average testing error norm based on eEBP training is 0.1243 and that based on the best EBP $(\epsilon = 0.5)$ training is 0.1395.

Furthermore, Elman's original sequence is modified as stated before in Example 2 and forms a context-free sequence [5] based on the same coding scheme and network structure. This context-free structure sets all strings of form $\omega\omega'$, that is, a string ω followed by a mirror-reversed copy ω' of ω, such that the original sequence of Example 2 becomes *bbaabb*, *diiiid*, and *guuuuuuug* (it is still a semirandom sequence since only the first three consonants are randomly generated). This is a context-free language and therefore needs a stack-like memory to keep track of its center-embedding recursion, which focuses on both cognitive and dynamical system aspects [5]. The generalization performance is shown in Fig. 15 and indicates that eEBP is better than the best EBP training with $\epsilon = 0.45$ as compared to that of the original sequence in Example 2, because the network needs to learn more with context-free input.

Example 3: A similar generator program is used as in [1] to construct a set of short (two- and three-word) sentences. As in Example 2, the input sequence was created in a semirandom fashion but is much closer to a real-time language prediction application because an online training/testing scheme is used, as follows.

The generator program used the 15 sentence templates in Table II to create 10 000 semirandom two- and three-word sentence frames. One word appropriate to each category (see Table III) was then randomly selected to fill in each frame. Each of these words was then replaced by a randomly assigned 31-bit vector, where each bit represented a different word. If the word was present, its corresponding bit was turned on. This scheme

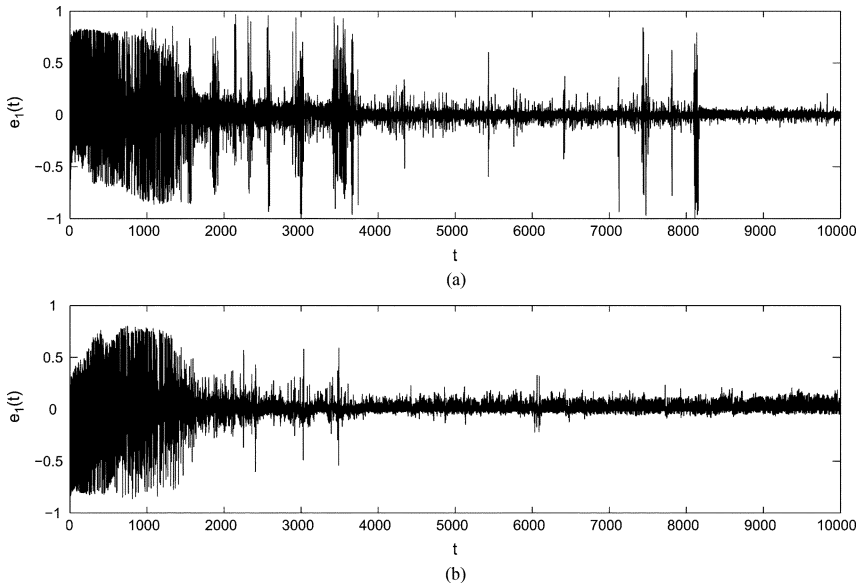

Fig. 6. Tracking of the bit-one prediction error $e_1(t)$ of Example 2. A comparison between (a) EBP ($\epsilon = 0.5$) and (b) eEBP.

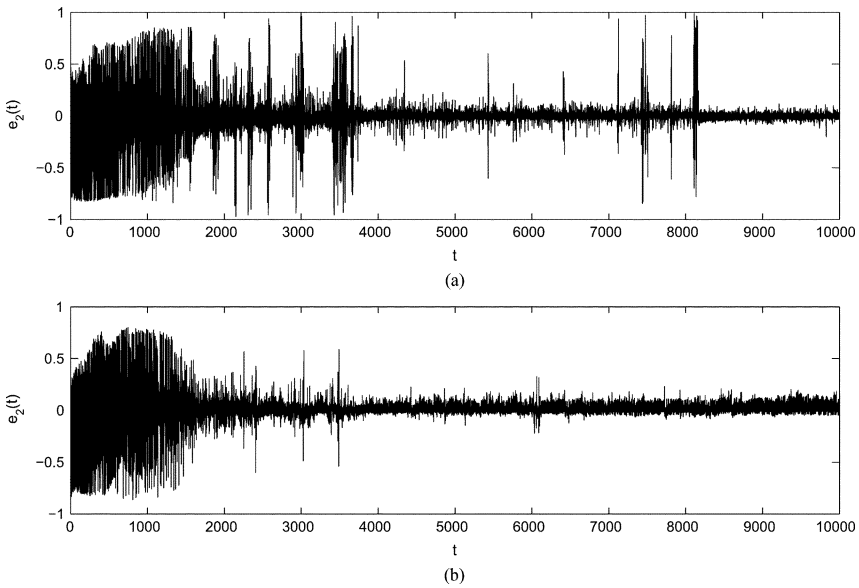

Fig. 7. Tracking of the bit-two prediction error $e_2(t)$ of Example 2. A comparison between (a) EBP ($\epsilon = 0.5$) and (b) eEBP.

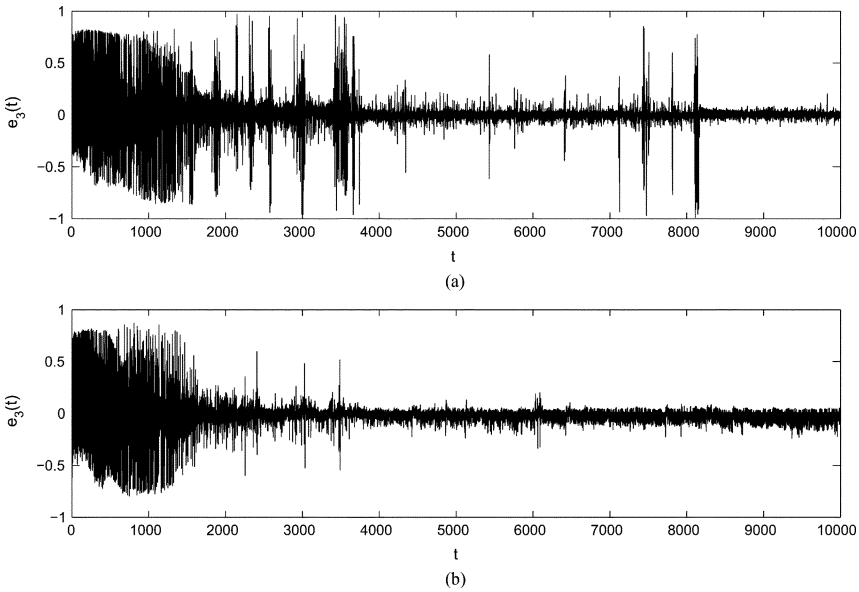

Fig. 8. Tracking of the bit-three prediction error $e_3(t)$ of Example 2. A comparison between (a) EBP ($\epsilon = 0.5$) and (b) eEBP.

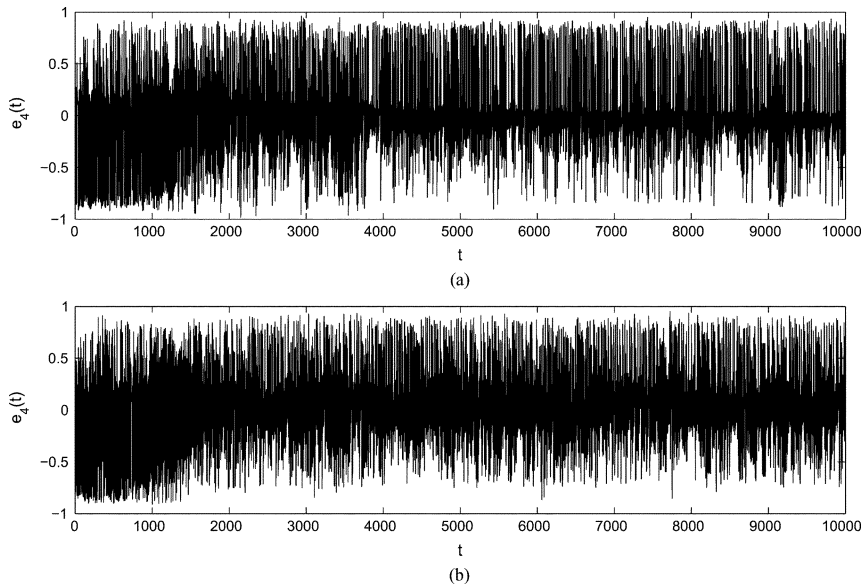

Fig. 9. Tracking of the bit-four prediction error $e_4(t)$ of Example 2. A comparison between (a) EBP ($\epsilon = 0.5$) and (b) eEBP.

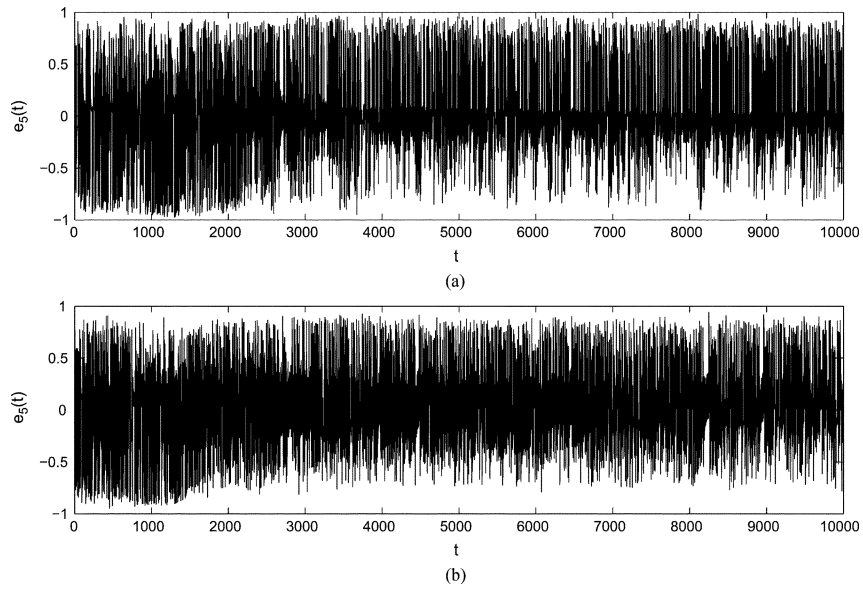

Fig. 10. Tracking of the bit-five prediction error $e_5(t)$ of Example 2. A comparison between (a) EBP ($\epsilon = 0.5$) and (b) eEBP.

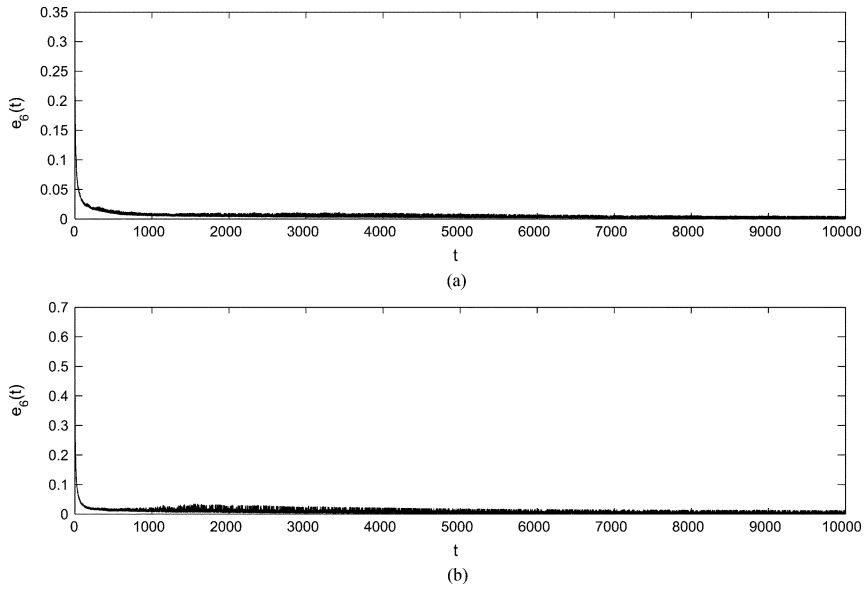

Fig. 11. Tracking of the bit-six prediction error $e_6(t)$ of Example 2. A comparison between (a) EBP ($\epsilon = 0.5$) and (b) eEBP.

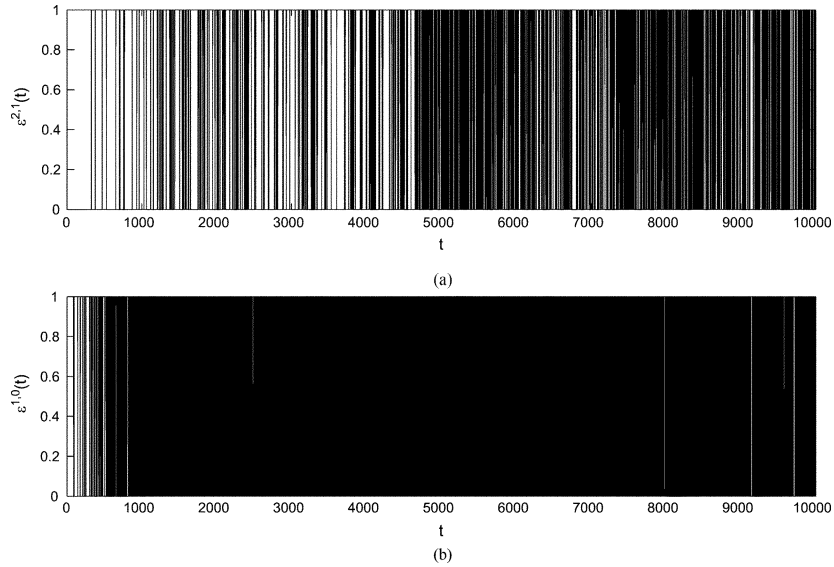

Fig. 12. Tracking of the eEBP training adaptive learning rates $\epsilon^{1,0}(t)$ and $\epsilon^{2,1}(t)$ for Example 2. (a) eEBP learning rate $e^{2,1}(t)$ of the output layer. (b) eEBP learning rate $e^{1,0}(t)$ of the hidden layer.

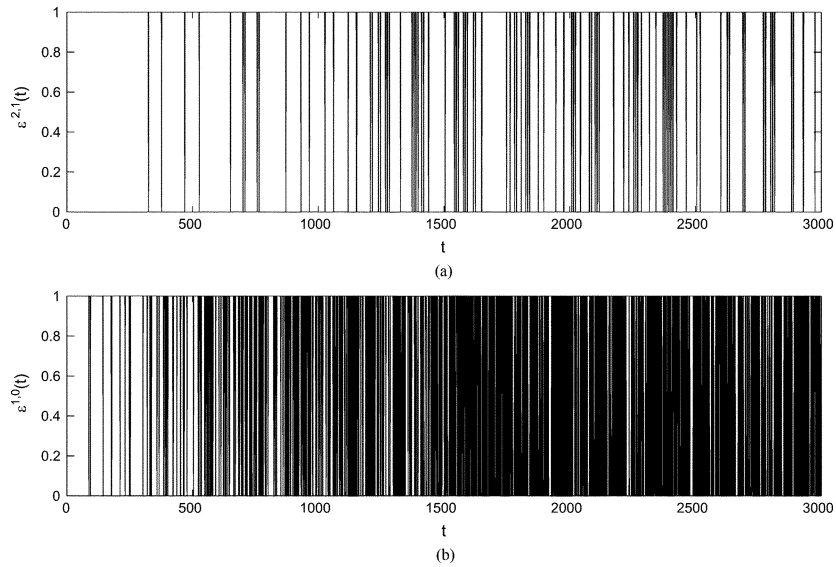

Fig. 13. Tracking of the eEBP training adaptive learning rates $\epsilon^{1,0}(t)$ and $\epsilon^{2,1}(t)$ for the first 3000 steps of Example 2. (a) eEBP learning rate $e^{2,1}(t)$ of the output layer. (b) eEBP learning rate $e^{1,0}(t)$ of the hidden layer.

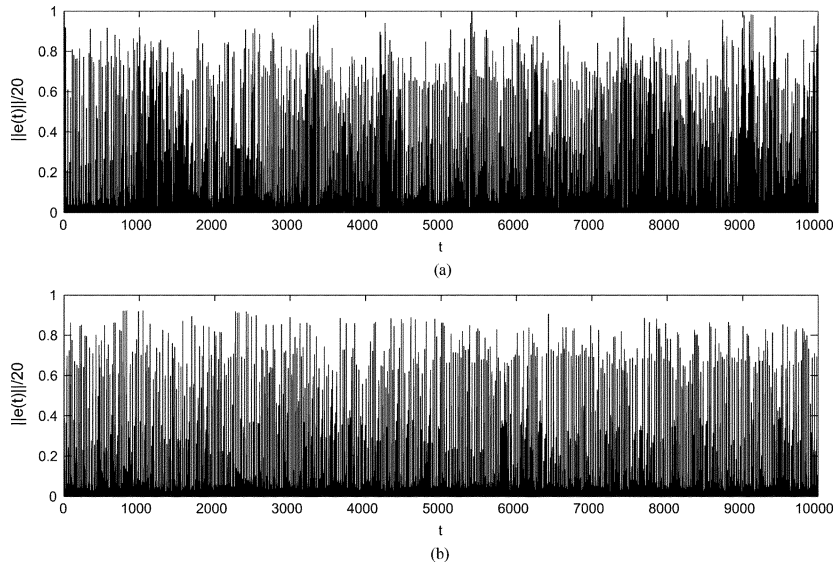

Fig. 14. Generalization performances of the (a) best EBP $(\epsilon = 0.5)$ and (b) eEBP training in terms of the average error norm $\|e(t)\|/20$ over 20 testing runs for Example 2.

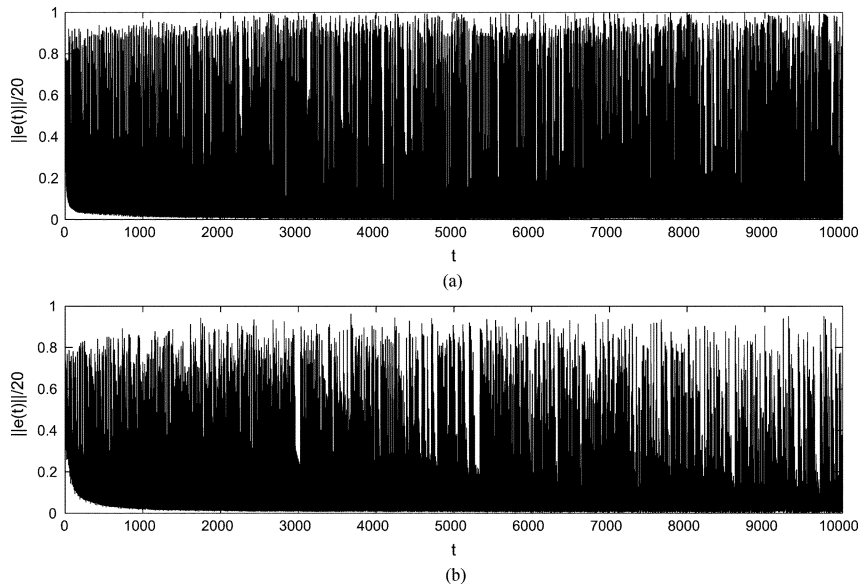

Fig. 15. Generalization performances of the (a) best EBP ($\epsilon = 0.45$) and (b) eEBP training in terms of the average error norm $\|e(t)\|/20$ over 20 testing runs for the context-free sequence input in Example 2.

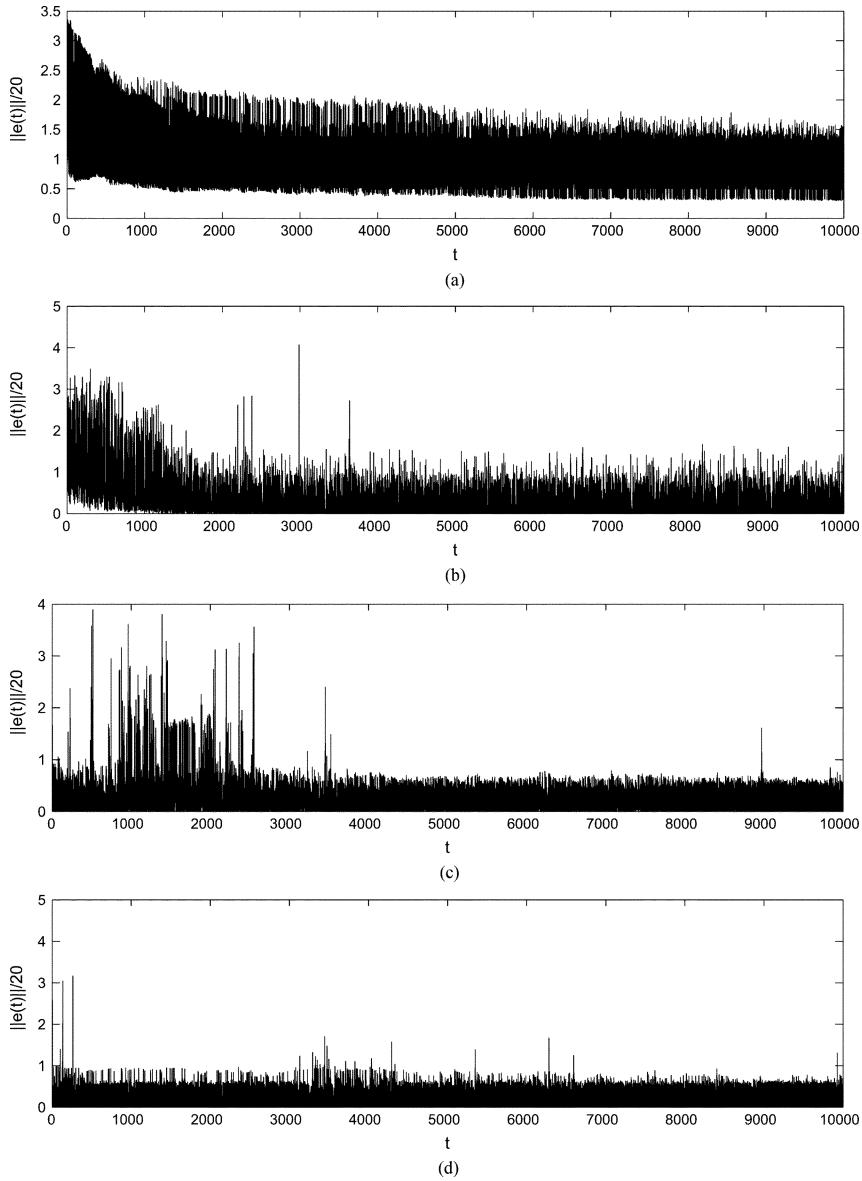

Fig. 16. Tracking of the error norms of training/testing for Example 3. (a) EBP ($\varepsilon = 1$). (b) EBP ($\varepsilon = 0.5$). (c) EBP ($\varepsilon = 0.1$) (d) eEBP.

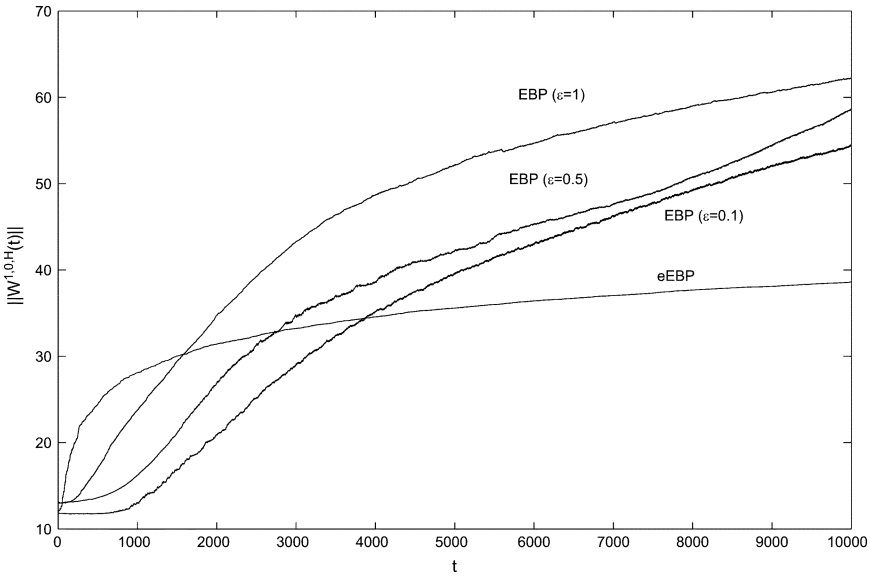

Fig. 17. Tracking of $\|W^{1,0,H}(t)\|$ norms between eEBP and EBP algorithms with different learning rates for Example 3.

TABLE II
SENTENCE GENERATOR TEMPLATES

WORD 1	WORD 2	WORD 3
NOUN-HUM	VERB-EAT	NOUN-FOOD
NOUN-HUM	VERB-PERCEPT	NOUN-INANIM
NOUN-HUM	VERB-DESTROY	NOUN-FAG
NOUN-HUM	VERB-INTRAN	
NOUN-HUM	VERB-TRAN	NOUN-HUM
NOUN-HUM	VERB-AGPAT	NOUN-INANIM
NOUN-HUM	VERB-AGPAT	
NOUN-HUM	VERB-AGPAT	
NOUN-HUM	VERB-DESTROY	NOUN-FRAG
NOUN-HUM	VERB-EAT	NOUN-HUM
NOUN-HUM	VERB-EAT	NOUN-ANIM
NOUN-HUM	VERB-EAT	NOUN-FOOD

TABLE III
CATEGORIES OF LEXICAL ITEMS USED IN SENTENCE SIMULATION

Category	Examples
NOUN-HUM	man, woman
NOUN-ANIM	cat, mouse
NOUN-INANIM	book, rock
NOUN-AGRESS	dragon, monster
NOUN-FRAG	glass, plate
NOUN-FOOD	cookie, bread
NOUN-INTRAN	think, sleep
NOUN-TRAN	see, chase
NOUN-AGPAT	move, break
NOUN-PERCEPT	smell, see
NOUN-DESTROY	break, smash
NOUN-EAT	eat

guaranteed mutual orthogonality of the vectors and conveyed no information about either the form class or the meaning of the words. Finally, each word vector was designed to be distinct and no breaks existed between successive sentences. The ENs used here are similar to those of the first two examples except that the input and output layers contained 31 nodes each and the hidden and context layers contained 100 nodes each. Note that this study did not select 150 hidden-layer nodes, as used in [1], because this large number may lead to a relatively larger training/testing error (see Remark 3 in Section III).

In terms of online training/testing, the ENs were to learn to predict the order of successive words; that is, training was integrated with testing. For a sequence of 10 000 words input in order, one at a time, the strategy was as follows. For each input cycle, the task was to predict the 31-bit vector that corresponded to the next word. At the end of the sequence, the process immediately restarted, with a new semirandomly selected first word and new initial random weights. Online training/testing continued in this fashion until the ENs experienced 20 complete passes through the sequence. Fig. 16 shows their generalization performance. It should be noted that the prediction task is nondeterministic and successive words cannot be predicted with certainty. An inevitable error is built into the process that reflects the generalization performance.

These were the steps of the so-called online training/testing procedure. The total sum (divided by 10 000 steps) of the average training/testing error norms based on eEBP training is 0.1672, compared to that of EBP, $0.2207(\epsilon = 0.1), 0.2963(\epsilon = 0.5), 0.9049(\epsilon = 1)$. Under these circumstances, it is suggested that the mean prediction error norm $\|e(t)\|/20$ can represent generalization performance with the minimum value achieved for the best algorithm through the 20 rounds of training/testing, while the optimal or suboptimal hidden-layer weighted norm was obtained by eEBP, as compared to EBP, as shown in Fig. 17.

V. CONCLUSION

This work proposes a new eEBP training algorithm with a MIMO adaptive dead zone scheme for ENs and proves weight convergence based on Lyapunov functions. The fully adaptive learning method is employed to optimize convergence speed and make an optimal or suboptimal tradeoff between real-time training and the generalization capabilities of the ENs to maximize learning speed and testing performance. Simulation experiments are carried out to verify the theoretical results and

clearly show that the learning performance of ENs based on eEBP training is better in terms of both weight convergence and generalization performance than that based on standard EBP training algorithms. This approach also agrees with previous research insofar as a relatively small weight matrix will lead to good generalization performance [9]. It is worth noting that the optimal or suboptimal solution for weight convergence in this paper implies that the adaptive dead zone approach be used to prevent the training error from getting too small according to the disturbance levels of individual outputs. Therefore, the overfitting problem can be avoided, to a certain extent, and relatively good generalization performance achieved [29]. This study also reveals that a relatively smaller network can result in better convergence and generalization performance, as noted in Remark 3. However, further research is needed to explore the theoretical link between a possible upper bound on the training error and the adaptive dead zone based on the structural risk minimization principle [29].

REFERENCES

[1] J. L. Elman, "Finding structure in time," *Cogn. Sci.*, vol. 14, pp. 179–211, 1990.

[2] M. Jordan, "Serial order: A parallel distributed processing approach," Inst. Cogn. Sci., Univ. California San Diego, San Diego, CA, ICS Rep. 8604, 1986.

[3] M. Jordan, "Constrained supervised learning," *J. Math. Psychol.*, vol. 36, pp. 396–425, 1992.

[4] X. Li, G. Chen, Z. Chen, and Z. Yuan, "Chaotifying linear Elman networks," *IEEE Trans. Neural Netw.*, vol. 13, no. 5, pp. 1193–1199, Sep. 2002.

[5] A. Gruning, "Stack-like and queue-like dynamics in recurrent neural networks," *Connection Sci.*, vol. 18, no. 1, pp. 23–42, 2006.

[6] S. C. Kremer, "On the computational power of Elman-style recurrent networks," *IEEE Trans. Neural Netw.*, vol. 6, no. 4, pp. 1000–1004, Jul. 1995.

[7] D. P. Mandic and J. A. Chambers, *Recurrent Neural Networks for Prediction: Learning Algorithms, Architecture and Stability*. Chichester, U.K.: Wiley, 2001.

[8] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 157–166, Mar. 1994.

[9] B. Hammer and P. Tino, "Recurrent neural networks with small weights implement definite memory machines," *Neural Comput.*, vol. 15, pp. 1897–1929, 2003.

[10] C.-M Kuan, K. Hornik, and H. White, "A convergence result for learning in recurrent neural networks," *Neural Comput.*, vol. 6, pp. 420–440, 1994.

[11] K. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Trans. Neural Netw.*, vol. 1, no. 1, pp. 4–27, Mar. 1990.

[12] C. F. Juang, C. T. Chiou, and C. L. Lai, "Hierarchical singleton-type recurrent neural fuzzy networks for noisy speech recognition," *IEEE Trans. Neural Netw.*, vol. 18, no. 3, pp. 833–843, May 2007.

[13] Q. Song, J. Spall, Y. C. Soh, and J. Ni, "Robust neural network tracking controller using simultaneous perturbation stochastic approximation," *IEEE Trans. Neural Netw.*, vol. 19, no. 5, pp. 817–835, May 2008.

[14] Q. Song, Y. L. Wu, and Y. C. Soh, "Robust adaptive gradient descent training algorithm for recurrent neural networks in discrete time domain," *IEEE Trans. Neural Netw.*, vol. 19, no. 11, pp. 1841–1853, Nov. 2008.

[15] J. F. Kolen, *A Field Guide to Dynamical Recurrent Networks*. New York: IEEE Press, 2001.

[16] S. Haykin, *Neural Networks*. Englewood Cliffs, NJ: Prentice-Hall, 1999.

[17] G. L. Plett, "Adaptive inverse control of linear and nonlinear systems using dynamic neural networks," *IEEE Trans. Neural Netw.*, vol. 14, no. 2, pp. 360–376, Mar. 2003.

[18] S. Koike, "A class of adaptive step size control algorithms for adaptive filters," *IEEE Trans. Signal Process.*, vol. 50, no. 6, pp. 1315–1326, Jun. 2002.

[19] R. Williams and J. Peng, "Gradient-based learning algorithms for recurrent neural networks," *Neural Comput.*, vol. 2, pp. 490–501, 1990.

[20] B. A. Pearlmutter, "Gradient calculations for dynamic recurrent neural networks: A survey," *IEEE Trans. Neural Netw.*, vol. 6, no. 5, pp. 1212–1228, Sep. 1995.

[21] M. Rupp and A. H. Sayed, "Supervised learning of perceptron and output feedback dynamic networks: A feedback analysis via the small gain theorem," *IEEE Trans. Neural Netw.*, vol. 8, no. 3, pp. 612–622, May 1997.

[22] J. Liang and M. M. Gupta, "Stable dynamic backpropagation learning in recurrent neural networks," *IEEE Trans. Neural Netw.*, vol. 10, no. 6, pp. 1321–1334, Nov. 1999.

[23] A. F. Atiya and A. G. Parlos, "New results on recurrent network training: Unifying the algorithms and accelerating convergence," *IEEE Trans. Neural Netw.*, vol. 11, no. 3, pp. 697–709, May 2000.

[24] Q. Song, J. Xiao, and Y. C. Soh, "Robust backpropagation training algorithm for multi-layered neural tracking controller," *IEEE Trans. Neural Netw.*, vol. 10, no. 5, pp. 1133–1141, Sep. 1999.

[25] A. C. Tsoi and A. D. Back, "Locally recurrent globally feedforward networks: A critical review of architectures," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 229–239, Mar. 1994.

[26] R. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural Comput.*, vol. 1, pp. 270–280, 1989.

[27] P. J. Werbos, "Generalisation of backpropagation with application to a recurrent gas market model," *Neural Netw.*, vol. 1, pp. 339–356, 1988.

[28] S. Hu and D. Liu, "On the global output convergence of a class of recurrent neural networks with time-varying inputs," *Neural Netw.*, vol. 18, pp. 171–178, 2005.

[29] V. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.

[30] G. Goodwin, *Adaptive Filtering Prediction and Control*. Englewood Cliffs, NJ: Prentice-Hall, 1984.

[31] Q. Song, "A robust information clustering algorithm," *Neural Comput.*, vol. 17, no. 12, pp. 2672–2698, 2005.

Qing Song (M'92) received the B.S. degree from Harbin Shipbuilding Engineering Institute, Harbin, China, in 1982, the M.S. degrees from Dalian Maritime University, Dalian, China, in 1986, and the Ph.D. degree from the Industrial Control Centre, Strathclyde University, Glasgow, U.K., in 1992.

He was an Assistant Professor at the Dalian Maritime University from 1986 to 1987. He worked as a Research Engineer at the System and Control Pte., Ltd., U.K., before joining the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, as a Lecturer in 1992. He is now an Associate Professor. He is the first inventor of a few neural network related patents and has authorized many referred international journal publications. He held short visiting research positions at The Johns Hopkins University, Baltimore, MD, and Tsinghua University, Beijing, China. He is also an active industrial consultant and the principal investigator of research programs targeted to industry in the area of computational intelligence.