Neurocomputing 87 (2012) 111–119

Contents lists available at SciVerse ScienceDirect

Neurocomputing

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

An evolving neuro-fuzzy technique for system state forecasting

Wilson Wang [a,*], De Z. Li [b], Joe Vrbanek [c]

[a] Mechanical Engineering, Lakehead University, Thunder Bay, Ont., Canada P7B 5E1
[b] Mechanical and Mechatronics Engineering, University of Waterloo, Waterloo, Ont., Canada N2L 3G1
[c] Project Management, Toronto Hydro One Inc., Toronto, Ont., Canada M5G 2P5

ARTICLE INFO

Article history:
Received 27 April 2011
Received in revised form
13 October 2011
Accepted 9 February 2012
Communicated by P. Zhang
Available online 25 February 2012

Keywords:
Evolving clustering
Neuro-fuzzy inference
System state forecasting
Machinery condition monitoring
Online training

ABSTRACT

A reliable predictor is very useful to real-world applications to forecast the future states of dynamic systems especially when their dynamic characteristics are time-varying. In this paper, an evolving neuro-fuzzy (eNF) technique is developed for system state forecasting. A novel clustering paradigm is developed for cluster generation and rule base modification. A new enhanced LSE method is proposed for online training of the eNF parameters, whose Lyapunov stability is verified theoretically. The effectiveness of the developed eNF predictor is evaluated based on simulation using some benchmark data sets. Next the eNF is implemented for the applications of currency exchange rate forecasting and machinery condition prognostics. Test results show that the enhanced LSE is computationally efficient and can improve reasoning convergence; the developed eNF predictor is an accurate forecasting tool. It can capture the system's dynamic behavior quickly and track the system's characteristics accurately. It is also a robust predictor to accommodate different system conditions.

1. Introduction

System state prognosis is a process to predict the future states of a dynamic system based on the available observations. A reliable predictor is very useful to a wide array of applications such as fatigue life prediction in materials, earthquake forecasting, predictive control, stock market prediction, and so on [1–3]. In engineering system condition monitoring, for example, the forecasting information can be used to provide an accurate alarm before fault reaches critical levels so as to prevent machinery performance degradation, malfunction, or even catastrophic failure [4,5].

Many methods have been proposed in the literature for time series prediction. The classical approaches are the use of stochastic models [6–8]; however, accurate analytical models are usually difficult to determine especially for complex dynamic systems. The alternative is the use of flexible models such as neural networks and neuro-fuzzy (NF) paradigms [9,10]. The authors' research team has also developed several adaptive NF schemes for machinery condition monitoring, in which the system parameters are trained online, whereas the forecasting reasoning structures remain fixed in operations or updated offline [11–13]. Although these techniques have been applied effectively in many applications, these NF predictors with fixed reasoning structures do not have sufficient adaptive capability to accommodate time-varying dynamic effects. One of the solutions to solve this problem is to use some clustering algorithm to evolve inference structures.

An evolving clustering system is a data-driven scheme that can adapt its structure and parameters simultaneously in a gradual and continuous way [14]. Since a fuzzy system is a universal function approximater and is capable of extracting interpretable knowledge, fuzzy paradigms have been utilized as a framework for designing evolving intelligent systems. For example, Angelov et al. proposed some evolving schemes based on the Takagi–Sugeno fuzzy model, namely eTS, for control and classification applications [15,16]; the cluster structures are determined by a potential measurement and consequent linear parameters are updated by least squares estimate (LSE) algorithm. Despite some promising results, one of the problems in these algorithms is that the spreads and/or centers of the newly generated clusters cannot be updated properly. Kasobov et al. suggested a transductive NF inference system with weighted data normalization (namely, TWNFI) for transductive reasoning operations [17]. But compared with eTS tools, TWNFI usually generates more rules in modeling nonlinear systems. Most currently proposed evolving paradigms applied clustering plus LSE for structure identification, without overall online training. Wang et al. proposed an evolving NF system for system classification [18]; however, the maximum number of clusters has to be predefined to limit the rule base dimension; furthermore, the optimization employs a cluster-based Jacobian

* Corresponding author. Tel.: +1 807 766 7174.
E-mail addresses: wilson.wang@lakeheadu.ca (W. Wang), d45li@uwaterloo.ca (D.Z. Li), joe.vrbanek@hydroone.com (J. Vrbanek).

0925-2312/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2012.02.006

matrix, the resulting training convergence is slow because input variable membership functions can be related to more than one cluster. The authors' group also proposed a clustering technique based on Euclidean distance measurement for system state forecasting application [19].

The main problems with the aforementioned techniques for prediction (and classification) are the required prior knowledge in selecting the ideal number of rules or number of nodes required for a certain task. In addition, most systems used LSE for online training, which has slow training convergence especially when errors become smaller. In order to tackle these problems and develop a more reliable predictor for real-time industrial applications, the objective of this work is to develop an evolving neuro-fuzzy (eNF) technique for system state forecasting (but it should be stated that the developed eNF technique can also be used for other applications such as pattern classification and control). The proposed eNF predictor is new in the following aspects: (1) a novel clustering algorithm is developed to effectively partition input–output spaces and exclude contradictory rules due to noise-affected clusters; (2) a novel enhanced LSE technique is proposed to adaptively optimize cluster parameters and to improve the convergence of the eNF predictor; and (3) the proposed eNF predictor is implemented for real-world forecasting applications, such as foreign currency exchange rate prediction and machinery condition monitoring.

The paper is organized as follows. A description of the proposed eNF clustering paradigm is described in Section 2. The proposed online training technique and its implementation are discussed in Section 3. The effectiveness of the developed eNF predictor is verified by simulation in Section 4. Some application examples are demonstrated in Section 5.

2. The eNF reasoning system

2.1. The NF reasoning model

Consider an input vector $\boldsymbol{x}=\{x_1, x_2,\ldots,x_n\}$. The fuzzy reasoning can be represented as

$$\textit{Rule } j:\ \text{IF } (x_1 \text{ is } A_1^j) \text{ AND}\ldots\text{AND } (x_n \text{ is } A_n^j) \text{ THEN } (y_j = q_j) \tag{1}$$

where $j\in[1,R]$, R is the total number of fuzzy rules (or clusters), A_i^j is the jth fuzzy set for x_i, $i\in[1,n]$; $y_j=[y_{j1},y_{j2},\ldots,y_{jM}]$ is an M-dimensional consequent (output) structure.

According to the consequent reasoning structure, fuzzy systems can be categorized into three types:

Type I (Mamdani model): $q_{jl}=B_{jl}$ where B_{jl} is a consequent fuzzy set, $l\in[1,M]$.
Type II (zero order TS model): $q_{jl}=a_{jl}$ where a_{jl} is a singleton, $l\in[1,M]$.
Type III (first order TS model): $q_{jl}=b_{0l}^j+b_{1l}^j x_1+\cdots+b_{nl}^j x_n$ where $b_\bullet^j$ are constants.

A Type-III model will be used in this work as an example to illustrate the forecasting reasoning of the proposed eNF system.

To facilitate the implementation of input/output partitioning [15,19], all the fuzzy set membership functions (MFs) in the proposed eNF are in the Gaussian form

$$\mu_{A_i^j}=\exp\left(-\frac{(x_i-m_{ij})^2}{2\sigma_{ij}^2}\right) \tag{2}$$

where m_{ij} and σ_{ij} are the centers and spreads of the MFs, respectively.

Despite the type of fuzzy reasoning structures, the premise rule structures of different fuzzy systems remain the same. If a max-product operator is used for the premise reasoning, the rule firing strength will be

$$\mu_j=\exp\left(-\sum_{i=1}^{n}\frac{(x_i-m_{ij})^2}{2\sigma_{ij}^2}\right),\quad j\in[1,\quad R] \tag{3}$$

After normalization of the rule firing strengths, the overall output will be

$$y=\sum_{j=1}^{R}\frac{\mu_j}{\mu_\sum}q_j,\quad j\in[1,\quad R] \tag{4}$$

where q_j is the result from the consequent part, and

$$\mu_\sum=\sum_{j=1}^{R}\mu_j=\sum_{j=1}^{R}\exp\left(-\sum_{i=1}^{n}\frac{(x_i-m_{ij})^2}{2\sigma_{ij}^2}\right) \tag{5}$$

2.2. The eNF clustering procedure

The developed eNF scheme evolves a number of clusters (fuzzy rules) based on incoming data in a gradual and continuous way. If the input/output data are clustered separately, it may result in some specific problems, for example, the generated clusters may not be consistent if some training data are noise affected. The proposed eNF technique clusters both input and output patterns simultaneously with the constraint of mapping consistency for the purpose of removing the noise affected data (or outliers). The cluster centers and spreads are updated so as to make the resulting clusters (rules) well-distributed over the input–output spaces. Fig. 1 shows the flowchart of the clustering processes of the suggested eNF, which are discussed as follows.

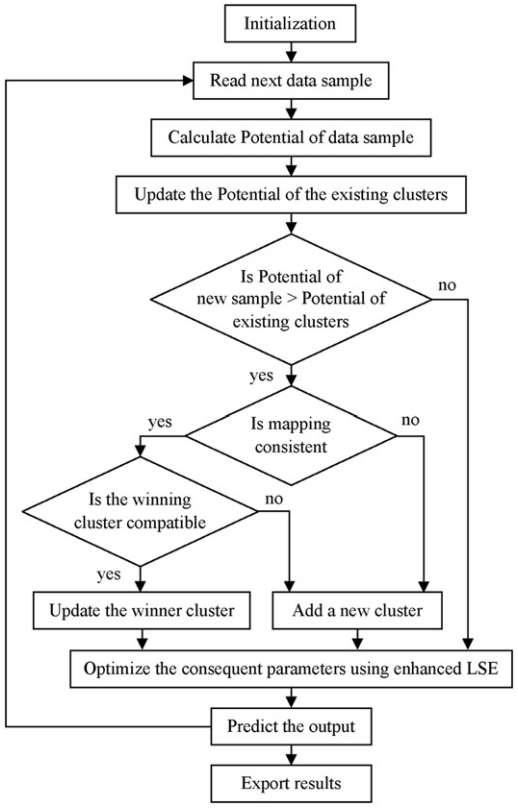


Fig. 1. Flowchart for the eNF clustering processes.

Step 1: *Initialize the system parameters*—The eNF scheme starts with an empty rule base. The first input data sample $\mathbf{z}_k=[\mathbf{x}_k, \mathbf{y}_k]$, $k\leftarrow 1$, defines the first rule (cluster) center, that is, $\mathbf{c}_k\leftarrow\mathbf{z}_k$. Then

$R\leftarrow 1$, $N_r\leftarrow 1$, $\mathbf{m}_R^I\leftarrow\mathbf{x}_k$, $\boldsymbol{\sigma}_R^I\leftarrow\mathbf{0}.25$, $\mathbf{m}_R^O\leftarrow\mathbf{y}_k$, $\boldsymbol{\sigma}_R^O\leftarrow\mathbf{0}.25$,

$P_k(\mathbf{z}_k)\leftarrow 1$, $P_k(\mathbf{c}_k)\leftarrow 1$,

where N_r is the number of samples in cluster r, $r\in[1,R]$ and R is the number of rules (clusters); $\mathbf{m}_R^I$, $\mathbf{m}_R^O$, $\boldsymbol{\sigma}_R^I$ and $\boldsymbol{\sigma}_R^O$ are the cluster centers and spreads in the input and output spaces, respectively. $P_k(\mathbf{z}_k)$is the potential of data sample $\mathbf{z}_k$, and $P_k(\mathbf{c}_k)$ is the potential of the center $\mathbf{c}_k$.

Step 2: *Compute the potential of the new data sample*—Input a next data sample, $\mathbf{z}_k=[\mathbf{x}_k, \mathbf{y}_k]$; $k\leftarrow k+1$. The potential of $\mathbf{z}_k$ is calculated by

$$P_k(\mathbf{z}_k)=\frac{k-1}{(k-1)(\boldsymbol{\theta}_k+1)+\boldsymbol{\sigma}_k-2\mathbf{v}_k} \tag{6}$$

where

$$\boldsymbol{\theta}_k=\sum_{i=0}^{n}(\mathbf{z}_k^i)^2;\ \boldsymbol{\sigma}_k=\boldsymbol{\sigma}_{k-1}+\sum_{i=0}^{n}(\mathbf{z}_{k-1}^i)^2;\ \mathbf{v}_k=\sum_{i=0}^{n}\mathbf{z}_k^i\boldsymbol{\beta}_k^i$$

and $\boldsymbol{\beta}_k^i=\boldsymbol{\beta}_{k-1}^i+\mathbf{z}_{k-1}^i$

$\boldsymbol{\beta}_k^i$ and $\boldsymbol{\sigma}_k$are initialized to the appropriate vector of zeros; n=dimension of the inputs $\mathbf{z}_k=[\mathbf{x}_k, \mathbf{y}_k]$.

Step 3: *Update the potential of existing clusters*—The recursive formula for updating the potential of all existing clusters at time instant k is given as

$$P_k(\mathbf{c}_r)=\frac{(k-1)P_{k-1}(\mathbf{c}_r)}{k-2+P_{k-1}(\mathbf{c}_r)+P_{k-1}(\mathbf{c}_r)\sum_{i=1}^{n+1}(\mathbf{d}_{k,k-1}^i)^2} \tag{7}$$

where $\mathbf{d}_{k,k-1}^i=\mathbf{z}_k^i-\mathbf{z}_{k-1}^i$, $\mathbf{c}_r$represents the x and y coordinates of all existing clusters, $r\in[1,R]$; $P_k(\mathbf{c}_r)$ is the potential of all existing clusters.

Step 4: *Winner cluster determination*—IF $P_k(\mathbf{z}_k)<P_k(\mathbf{c}_r)$, or the potential of the current data point is less than the potential of all existing clusters, THEN go to Step 6 and the consequent parameters are updated.

Otherwise, IF $P_k(\mathbf{z}_k)\geq P_k(\mathbf{c}_r)$, THEN determine the wining cluster:

The winner cluster in the input space is as follows:

The winner cluster in the input space is : $W_I=\arg\min_{k=1}^{K}\|\mathbf{m}_k^I-\mathbf{x}_k\|$

The winner cluster in the output space : $W_O=\arg\min_{k=1}^{K}\|\mathbf{m}_k^O-\mathbf{y}_k\|$

where $k\in[1,K]$, and K is the total number of data pairs.

Step 5: *Recognize the structure*—IF $W_I=W_O$ and the new data sample has a descriptive grade $d_{W,k}=\|\mathbf{m}_W-\mathbf{m}_k\|<\mu_{min}$, THEN merge the new data set to the winning cluster. The section of μ_{min} depends on applications. A smaller μ_{min} will result in more clusters and more computation burden, while a larger μ_{min} will generate fewer clusters but cluster discriminative property is degraded. By some simulation tests, $\mu_{min}=0.3$ is selected in this case.

The winner cluster parameters are updated in the input space and output space, respectively, whereas the other cluster information remains unchanged:

$$(\boldsymbol{\sigma}_{W,k}^I)^2\leftarrow(\boldsymbol{\sigma}_{W,k-1}^I)^2+\frac{1}{N_W}\left[(\mathbf{x}_k-\mathbf{m}_{W,k-1}^I)^2-(\boldsymbol{\sigma}_{W,k-1}^I)^2\right]$$

$$\mathbf{m}_{W,k}^I\leftarrow\mathbf{m}_{W,k-1}^I+\frac{\mathbf{x}_k-\mathbf{m}_{W,k-1}^I}{N_W}$$

$$(\boldsymbol{\sigma}_{W,k}^O)^2\leftarrow(\boldsymbol{\sigma}_{W,k-1}^O)^2+\frac{1}{N_W}\left[(\mathbf{y}_k-\mathbf{m}_{W,k-1}^O)^2-(\boldsymbol{\sigma}_{W,k-1}^O)^2\right]$$

$$\mathbf{m}_{W,k}^O\leftarrow\mathbf{m}_{W,k-1}^O+\frac{\mathbf{y}_k-\mathbf{m}_{W,k-1}^O}{N_W}$$

Otherwise, IF $W_I\neq W_O$, or there is no winning cluster or IF the description of the new data sample $d_{W,k}\geq\mu_{min}$, THEN create a new cluster:

$R\leftarrow R+1$, $N_R\leftarrow 1$, $\mathbf{m}_R^I\leftarrow\mathbf{x}_k$, $\boldsymbol{\sigma}_R^I\leftarrow\mathbf{0.25}$; $\mathbf{m}_R^O\leftarrow\mathbf{y}_k$, $\boldsymbol{\sigma}_R^O\leftarrow\mathbf{0.25}$.

These criteria are applied to prevent contradictory rules to exclude those noise affected clusters. For example, two closest clusters may not be merged to one cluster if they belong to different classes (out of consistency).

Step 6: *Tune the consequent parameters online*—The optimization is taken by the use of the proposed enhanced LSE to be discussed in Section 3.

Step 6: *Output prediction and parameter optimization*—At this stage, the output is predicted via Eq. (4).

Proceed back to Step 2. The corresponding results can be exported for data analysis.

When all the data samples have been inputted to the system (i.e., $k=K$), end the program and perform offline training to optimize both the premise and consequent cluster parameters, as discussed in Section 3.

3. Training of the evolving NF predictor

Once a new data sample is input to the eNF scheme, the clusters are created based on the proposed eNF clustering technique as discussed in Section 2. Its consequent parameters will be optimized by online training using the proposed the enhanced LSE.

3.1. Online training based on the enhanced LSE

The enhanced LSE technique is proposed in this work to recognized and update the linear parameters of the eNF. As the kth data sample $\mathbf{z}_k$ is provided to the system, the cluster centers and spreads are determined based on the proposed eNF clustering paradigm. The prediction of Eq. (4) can be expressed as $y_{k+1}=\boldsymbol{\Omega}_k^T\boldsymbol{\theta}_k$, where $\boldsymbol{\Omega}_k$ is the resulting matrix from the fuzzy inference operations, which is related to those nonlinear parameter; and $\boldsymbol{\theta}_k$ is the vector of the consequent linear parameters to be determined. The update law based on the classical LSE is represented as

$$\tilde{\boldsymbol{\theta}}_k=\boldsymbol{\theta}_k-\boldsymbol{\theta}_{k-1}=\boldsymbol{\Psi}_k\boldsymbol{\Omega}_k\mathbf{e}_k \tag{8}$$

where

$$\boldsymbol{\Psi}_k=\boldsymbol{\Psi}_{k-1}-\frac{\boldsymbol{\Psi}_{k-1}\boldsymbol{\Omega}_k\boldsymbol{\Omega}_k^T\boldsymbol{\Psi}_{k-1}}{1+\boldsymbol{\Omega}_k^T\boldsymbol{\Psi}_{k-1}\boldsymbol{\Omega}_k} \tag{9}$$

$$\mathbf{e}_k=\mathbf{y}_k^d-\mathbf{y}_k \tag{10}$$

$$\mathbf{y}_k=\boldsymbol{\Omega}_k^T\boldsymbol{\theta}_{k-1} \tag{11}$$

$\boldsymbol{\Psi}_k$ and $\boldsymbol{\Psi}_{k-1}$ are $n\times n$ covariance matrices at sample (or time) instants k and $k-1$, respectively, and n is the horizon of the input space (or the number of linear parameters). In the above expressions, $\tilde{\boldsymbol{\theta}}_k=\boldsymbol{\theta}_k-\boldsymbol{\theta}_{k-1}$ is the increment of linear parameter vector in each update step; $\boldsymbol{\theta}_k$ and $\boldsymbol{\theta}_{k-1}$ are the vectors of linear parameters at instants k and $k-1$. $\mathbf{y}_k^d$ is the desired output corresponding to $\boldsymbol{\Omega}_k$ at time instant k.

In fact, the update step $\tilde{\boldsymbol{\theta}}_k$ in Eq. (8) becomes very slow as the error $\boldsymbol{e}$ approaches zero in the classical LSE. The enhanced LSE method is proposed to solve this problem. A T-function is suggested

in this work to speed up convergence, which is defined as

$$T(e)=\begin{cases}\frac{1}{1+\exp[-\omega(e-\nu)]} & \text{if } e>0\\ 0 & \text{if } e=0\\ -\frac{1}{1+\exp[\omega(e+\nu)]} & \text{if } e<0\end{cases} \tag{12}$$

where $\omega \geq 0$ and $\nu \geq 0$ are constants.

T-function aims to bind the acceleration and to speed up convergence. Different ω and ν values result in different shapes and locations of the T-function. $\nu=0$ can be used for general search applications when the objective is to approach zero (i.e., errors). ω should be tuned to adjust the slop of the T-function as the error approaches zero. By some simulation tests, $\{\omega=15, \nu=0\}$ are selected in this case, the corresponding T-function is shown in Fig. 2(a). The convergence acceleration effect of T-function is illustrated in Fig. 2(b). It can be seen that the convergence speed is improved by T-function as states approach the equilibrium over the range of $e \in [-0.2,0.2]$. It should be stated that the suggested T-function can also be used for other applications in classification and control by revising the related parameters.

With the T-function, the parameter update Eq. (8) is revised as

$$\tilde{\boldsymbol{\theta}}_k = \boldsymbol{\Psi}_k \boldsymbol{\Omega}_k[\mathbf{e}_k + T(\mathbf{e}_k)] \tag{13}$$

Comparing Eqs. (13) and (8), the increment of target vector $\tilde{\boldsymbol{\theta}}_k$ is enlarged if T-function is applied especially when $\mathbf{e}$approaches to zero.

The Lyapunov stability of the proposed enhanced LSE is proven as follows. Assume that the initial conditions are $\boldsymbol{\theta}_0 = \mathbf{0}$ and $\boldsymbol{\Psi}_0 = \rho \mathbf{I}$, where $\boldsymbol{I}$ is an identity matrix; ρ is a sufficiently large positive number so as to guarantee that $\boldsymbol{\Psi}_k$ is positive semi-definite over a sufficiently long period ($\rho = 10^3$ in this case). According to the definition of positive semi-definite, it is seen that $\boldsymbol{\Omega}_k^T \boldsymbol{\Psi}_k \boldsymbol{\Omega}_k \geq 0$ for all non-zero vectors $\boldsymbol{\Omega}_k$ with real entries.

Calculate the derivative of Eq. (11),

$$\dot{\mathbf{y}}_k = \boldsymbol{\Omega}_k^T \dot{\boldsymbol{\theta}}_{k-1} = \boldsymbol{\Omega}_k^T \frac{\boldsymbol{\theta}_k - \boldsymbol{\theta}_{k-1}}{\Delta t} = \boldsymbol{\Omega}_k^T \boldsymbol{\Psi}_k \boldsymbol{\Omega}_k[\mathbf{e}_k + T(\mathbf{e}_k)]\Delta t^{-1} \tag{14}$$

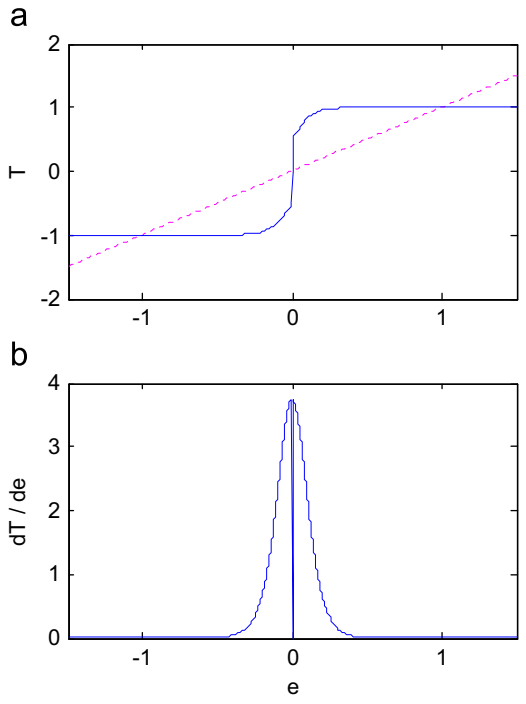


Fig. 2. (a) The graph of $T(e)$ (solid line) and $y=e$ (dashed line) when $\omega=15$ and $\nu=0$; (b) graph of function $dT(e)/de$.

$$\dot{\tilde{\mathbf{y}}} = \dot{\mathbf{y}}_k - \dot{\mathbf{y}}_k^d = \dot{\mathbf{y}}_k = \boldsymbol{\Omega}_k^T \boldsymbol{\Psi}_k \boldsymbol{\Omega}_k[\mathbf{e}_k + T(\mathbf{e}_k)]\Delta t^{-1} \tag{15}$$

If the Lyapunov function is selected as

$$V = \tfrac{1}{2}\tilde{\mathbf{y}}^T \tilde{\mathbf{y}}, \tag{16}$$

its derivative becomes

$$\begin{aligned}\dot{V} &= \tilde{\mathbf{y}}^T \dot{\tilde{\mathbf{y}}} = \tilde{\mathbf{y}}^T \boldsymbol{\Omega}_k^T \boldsymbol{\Psi}_k \boldsymbol{\Omega}_k[\mathbf{e}_k + T(\mathbf{e}_k)]\Delta t^{-1}\\ &= -\mathbf{e}_k^T \boldsymbol{\Omega}_k^T \boldsymbol{\Psi}_k \boldsymbol{\Omega}_k[\mathbf{e}_k + T(\mathbf{e}_k)]\Delta t^{-1}\\ &= |\boldsymbol{\Omega}_k^T \boldsymbol{\Psi}_k \boldsymbol{\Omega}_k|(-|\mathbf{e}_k|^2 - |\nu|)\Delta t^{-1} \leq 0\end{aligned} \tag{17}$$

Then it is concluded that the proposed enhanced LSE method is Lyapunov stable. Furthermore, since the enhanced LSE can reduce the search space dimension of the backpropagation, the overall training speed can be improved.

3.2. Hybrid offline training process

The training the developed eNF predictor is performed by two processes, online training using the enhanced LSE technique and offline training. Offline training is performed as long as all the data samples have been inputted to the eNF scheme to optimize both the cluster parameters and the consequent parameters, as discussed in this section. Currently there are many training algorithms proposed in the literature. For example, evolutionary computation methods such as the genetic algorithm (GA), as well as its associated methods (e.g., the Laplace crossover, real-coded GA, and parent-centric GA) have been proposed to reduce local minima [20,21]. GA is a derivative-free optimization method that has several advantages over other classical optimization algorithms (e.g., the gradient methods) such as parallel-processing and flexible search space. However, GA-based methods usually converge slowly especially when the search spaces become complex [22].

To speed up training convergence for real-time forecasting applications, a hybrid training technique is adopted in this work to update the eNF predictor offline. The premise and consequent parameters of the eNF scheme are optimized separately. Each training epoch takes two runs: in the forward pass, the consequent linear parameters are optimized by the proposed enhanced LSE technique; in the backward pass, the premise parameters will be optimized using the adaptive Levenberg–Marquardt (L–M) method as we suggested in [19]. Since each training epoch consists of two independent learning processes which take different initial conditions, it is possible to escape more local minima in training.

In addition, as stated in [19], the adaptive L–M algorithm possesses quadratic convergence close to a minimum. Its convergence property is still reasonably well even if the initial estimates are relatively poor. In addition, the L–M algorithm has been proven globally convergent in many applications by properly choosing the step factors. On the other hand, adaptive training is preferred in real-time applications because of the following: (1) it is necessary for time-varying systems; (2) it can extract knowledge from new data and perform a higher level adaptation of model parameters; (3) it possesses randomness that may help to escape from a local minimum; and (4) it is useful when the number of training data is large.

4. Performance evaluation by simulations

The performance of the proposed eNF predictor is evaluated in this section by two commonly used benchmark data sets before it is implemented for real-world applications.

4.1. Mackey–Glass data forecasting

The Mackey–Glass data series is commonly used to evaluate forecasting schemes due to its specific natures such as chaotic, non-periodic, and non-convergence, which is defined as [23]

$$\frac{dx(t)}{dt} = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t) \tag{18}$$

A data set, with initial conditions $x(0)=1.2$, $x(0)=0$ when $t<0$, $dt=1$ and, $\tau=30$ is used in this case for performance evaluation. 1600 data pairs are selected: the first 800 samples are used for training whereas the remaining 800 samples for testing the identified model. In this work, four input variables are used in the predictor $\{x_0, x_{-s}, x_{-2s}, x_{-3s}\}$, corresponding to the current and previous time steps, where s is the step horizon. To make a comparison, the results from the related and well known forecasting tools such as feedforward neural network (FNN), radial basis function (RBF) network, ANFIS [10], eTS [15], and TWNFI [17] and are also included. The consequent linear parameters in these predictors will be trained by the classical LSE method.

To examine the effectiveness of the proposed enhanced LSE training technique, the test results of the eNF predictor trained by the classical LSE method (namely, eNF-0) and the enhanced LSE technique (namely, eNF) are also listed. The averaged ten-steps-ahead (i.e., $s=10$) forecasting results of x_{+s} are summarized in Table 1.

Each input variable in the ANFIS predictor has two MFs (*small* and *large*), which results in 16 fuzzy rules for this forecasting application (with 96 parameters to be updated). The ANFIS has a fixed fuzzy structure and optimization is for its parameters only (gradient method for the premise parameters and LSE for consequent parameters). To make a comparison with some classical predictors, the FNN has 13 neurons (4-8-1) with 106 parameters, and the RBF predictor has 13 neurons with 102 network parameters. These parameters are trained by the classical gradient algorithm.

From Table 1, it is seen that the ANFIS predictor performs better those the classical predictors based on FNN and RBF, due to its adaptive fuzzy reasoning operations. However, the accuracy of these three predictors with fixed architectures is lower than those evolving predictors.

After 100 offline training epochs, the TWNFI and eTS predictors have generated 8 and 6 clusters (rules), respectively. Compared eNF-0 with the three classical predictors with similar training algorithms (LSE), the eNF clustering approach can effectively reduce the number of clusters (or fuzzy rules) and improve the processing accuracy.

The proposed eNF predictor generates only four clusters in this test. Compared eNF with eNF-0, the proposed enhanced LSE technique can reduce the search space dimension of the back-propagation and improve the overall training speed. It can also effectively improve the mapping accuracy of the fuzzy reasoning and the convergence of the space searching especially as the error becomes smaller. Fig. 3 shows the ten-steps-ahead forecasting result by the developed eNF predictor. Fig. 4 demonstrates a comparison of the convergence speed of eNF-0 and eNF.

Table 1
Forecasting results (10-steps-ahead) of a Mackey–Glass data set (RMSE: Root Mean Squares Error).

Forecasting schemes	No. of clusters	Average RMSE ($\times 10^{-2}$)	Average training time per epoch (s)
BNN	–	11.71	0.23
RBF	–	8.42	0.40
ANFIS	16	5.86	0.26
TWNFI	8	4.65	0.39
eTS	5	3.18	0.36
eNF-0	5	2.93	0.38
eNF	4	2.07	0.32

4.2. Sunspot activity data prediction

The sunspot activity record is another commonly used benchmark data set in the research of time series forecasting, due to its specific natures such as nonlinear, non-Gaussian, and non-stationary. The available data set contains the annual sunspot activity record from years of 1700 to 2010, with a mean of 49.494 and standard deviation of 40.453 [24]. The first 230 samples (the years 1700–1929) are used to train the predictors, whereas the remaining data pairs are used to test the identified models. Table 2 summarizes the forecasting results for a one-step-ahead ($s=1$) prediction (the test results using the classical FNN and RBF predictors are not listed due to their poor performance in this case). The ANFIS cannot capture the system dynamic behavior effectively in this case because its fixed reasoning structure will limit its adaptive capability.

The eNF-0 generates 4 rules (clusters) fewer than both the TWNFIS (7 rules) and eTS (6 rules) due to its effective clustering operations. Although both the eNF and eNF-0 predictors create

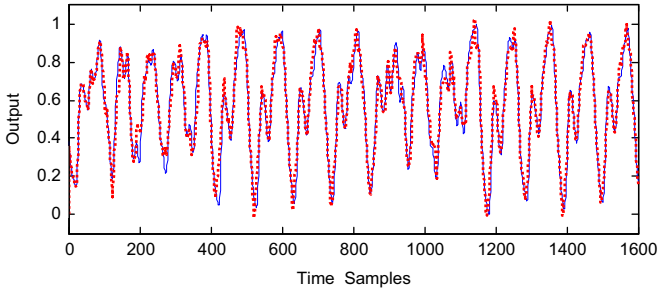


Fig. 3. Forecasting results (ten-steps-ahead) of a Mackey–Glass data set (solid lines) using the eNF predictor (dotted line).

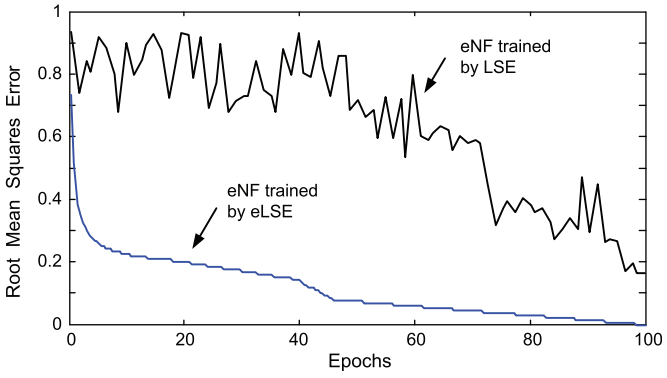


Fig. 4. Comparison of convergence speed of the eNF predictors, trained by enhance LST (eNF) and the classical LST (eNF-0).

Table 2
Forecasting results (one-step-ahead) of a sunspot activity record.

Forecasting schemes	Number of clusters	Average RMSE ($\times 10^{-2}$)	Average training time per epoch (s)
ANFIS	16	175.35	0.08
TWNFI	7	19.86	0.15
eTS	6	13.73	0.12
eNF-0	4	10.84	0.16
eNF	4	7.08	0.11

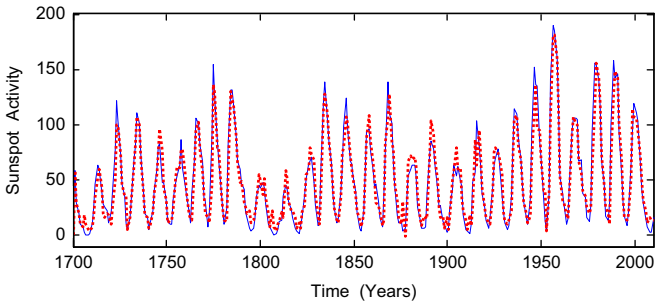


Fig. 5. Forecasting results (one-step-ahead) of the sunspot activity record (solid line) and the predicted values using the eNF predictor (dotted line).

Table 3
Forecasting results (one-step-ahead) of US-CAN currency exchange rate.

Forecasting schemes	No. of clusters	Average RMSE ($\times 10^{-3}$)	Average training time per epoch (s)
TWNFI	5	5.24	0.36
eTS	5	5.07	0.29
eNF-0	4	4.68	0.34
eNF	3	2.61	0.22

4 rules in this case, the eNF scheme significantly outperforms the eNF-0 predictor in terms of forecasting accuracy and convergence efficiency due to the application of the enhanced LSE technique. The forecasting result by the eNF predictor is shown in Fig. 5.

5. Application examples

In this section, two examples are used to illustrate the real-world applications of the developed eNF predictor: one is for foreign currency exchange rate prediction, and another is for machinery health condition prognosis. The related predictors as illustrated in Section 4.2, except for the ANFIS, are used in the following application examples as a comparison.

5.1. Foreign currency exchange rate prediction

The first application example is to predict the average daily exchange rate between the US dollars (USD) and the Canadian dollars (CAD). The available data sets are from the online database of the PACIFIC Exchange Rate Service [25]. The data selected are over the past 4 years from April 2, 2006 to March 1, 2011, excluding weekends and holidays. In total, there are 982 data samples, with a mean of $0.941 and standard deviation of $0.065. The first 442 data pairs (from April 2, 2006 to December 31, 2009) are used for training, and the remaining data samples (from January 2, 2009 to March 1, 2011) are used for testing the related predictors. Table 3 summarizes the one-step-ahead ($s=1$) forecasting results using the related predictors. It is seen that the eNF predictor can provide accurate forecast results (i.e., the second day exchange rate), in terms of efficiency (3 rules), accuracy (RMSE=0.00261) and convergence (0.22 s per epoch). Fig. 6 shows the forecasting result using the eNF predictor, and Fig. 7 illustrates the recognized fuzzy reasoning model.

5.2. Machinery health condition prognosis

5.2.1. Experimental setup and signal processing

Most machinery faults are related to transmission systems, such as gears and bearings. A bearing defect deteriorates transmission accuracy and increases noise level; On the other hand, a gear fault can not only decrease transmission accuracy but also cause unexpected failures. As a result, a few examples are given in this work to demonstrate the application of the developed eNF predictor for gear fault condition monitoring.

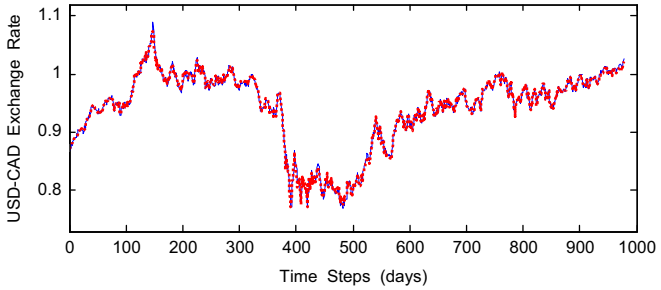


Fig. 6. Forecasting results (one-step-ahead) of the USD-CAD exchange rate: real values (solid line) and the predicted values using the eNF predictor (dotted line).

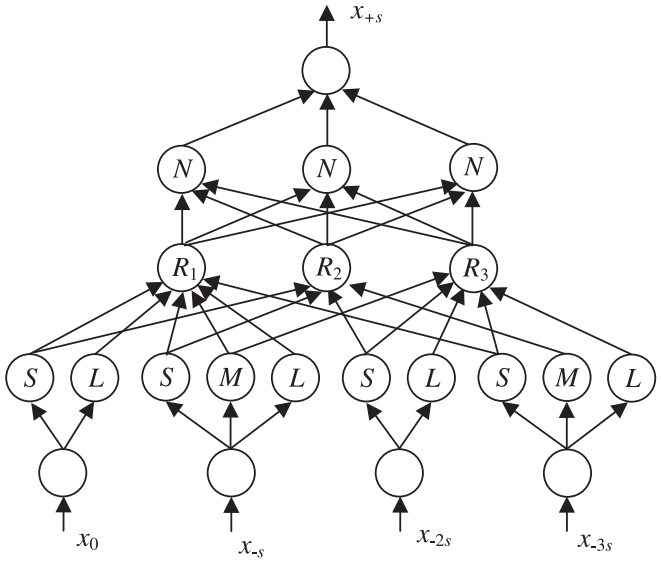


Fig. 7. The identified eNF model from the USD-CAD exchange rate forecasting.

Fig. 8 shows the experimental setup used in this study. The apparatus consists of two -1 HP permanent magnet DC motors and a single stage gearbox. A pair of spur gears with 14 and 16 teeth are tested. The purpose to use relatively small gears for testing in this work is to facilitate fault propagation process. The motors and gearbox are mounted onto a stiffened I-beam which is anchored to a massive concrete block. The speed controller allows gearbox operation in the range of 20–1400 rpm. The speed of the drive motor and the resistor network can be adjusted continuously to accommodate the range of speed/torque operating conditions. The vibration is measured with an accelerometer, mounted on the gearbox housing in the direction of gear action. An optical sensor is mounted in proximity to a slotted disk attached to the driving shaft, which provides a one-pulse-per-revolution signal to be used for the synchronous average filtering operation. The signals from both sensors are properly preprocessed (i.e., signal amplification, adjustment, anti-aliasing filtering, and A/D transmission) and then fed to a computer for further processing.

In this work, the condition monitoring of the gear system is conducted gear by gear. The measured vibration from the experimental setup, however, is an overall signal associated with different vibratory sources, such as shafts, bearings, gear mesh, and motors. Each rotary component will contribute a vibratory signal with specific spectral characteristics. Each gear signal can

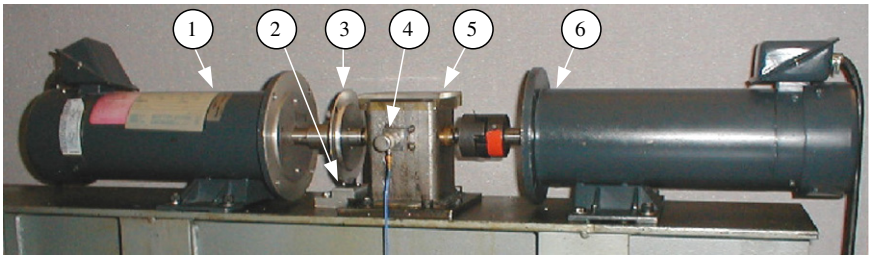


Fig. 8. Experimental setup: 1—drive motor, 2—optical sensor, 3—slotted disk, 4—accelerometer, 5—gearbox, 6—load motor.

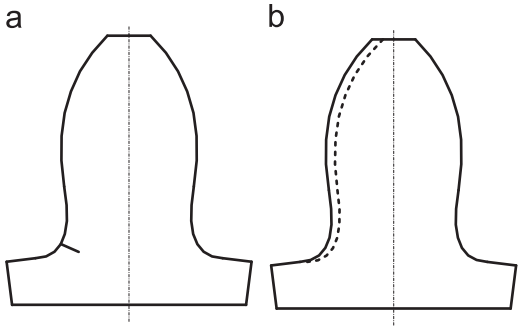


Fig. 9. Gear conditions tested: (a) cracked gear and (b) worn gear.

be differentiated by applying a time synchronous filtering process [13]. As a result, the signals nonsynchronous to the rotation of the gear of interest can be removed.

In condition monitoring, the monitoring indices should be sensitive to pattern modulation due to machinery fault but insensitive to noise. Several signal processing techniques have been proposed in the authors' research group for gear system condition monitoring. In this case, a beta kurtosis monitoring index is used as an example for gear fault diagnosis/prognosis. Details related to signal processing and derivation of the related monitoring index can be found in [13]. The predictors are applied in this case to forecast the future values of this monitoring index.

Two typical gear faults, fatigue cracking and wear, are tested in this study as represented in Fig. 9. They correspond to the two common types of defects in gear systems, localized and distributed faults.

5.2.2. Cracked gear monitoring

At first, the gear with 14 teeth is healthy, but the gear with 16 teeth has a transverse cut with 10% of the tooth root thickness to simulate an initial gear crack. The tests are conducted under load levels from 0.1 to 1 hp and motor speeds from 20 to 1400 rpm. During testing, motor speed and load levels are randomly changed to simulate time-varying operating conditions. The four evolving predictors are implemented for testing. All the predictors are trained by an old data set corresponding to a scoring gear defect. The monitoring time-step is set at $s=3$. The time horizon of monitoring is selected as 20 min, that is, the related predictors are applied automatically in every 20 min to forecast the upcoming values of the beta kurtosis index for the gear of interest (i.e., the gear with 16 teeth) after three steps, or 60 min (or 1 h). The test continues until the cracked tooth is broken off about 130 h later. The forecasting results from different predictors are listed in Table 4.

It is seen that eNF performs effectively in this test, which has recognized 5 clusters, whereas 6 clusters are created by eNF-0 which is trained by the classical LSE. TWNFI and eTS models generate 9 and 7 clusters, respectively. Fig. 10 shows the

Table 4
Forecasting results (three-steps-ahead) of cracked gear.

Forecasting schemes	No. of clusters	Average RMSE ($\times 10^{-3}$)	Average training time per epoch (s)
TWNFI	9	2.42	0.16
eTS	7	3.17	0.15
eNF-0	6	1.82	0.16
eNF	5	1.35	0.11

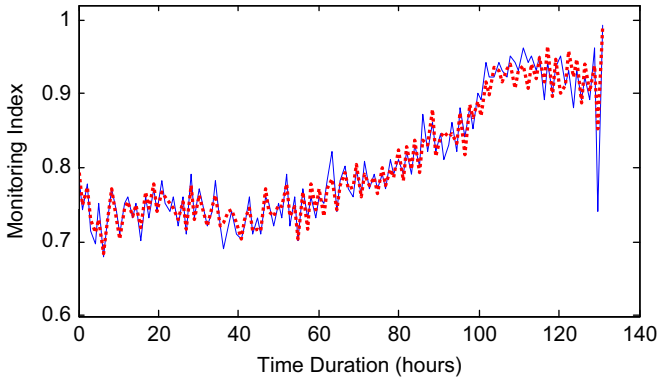


Fig. 10. Forecasting results (three-steps-ahead) of the cracked gear: real values (solid line) and the predicted values using the eNF predictor (dotted line).

forecasting results of the eNF paradigm. It is seen that the eNF predictor can capture system's dynamic behaviors and track their propagation trend effectively. During the last monitoring section, big fluctuations appear because the gear mesh dynamics change dramatically just before and after the tooth failure. The eNF predictor has provided alarms about 8–10 samples prior the tooth breakage. This information is a very valuable indicator for gear system condition monitoring. Fig. 11 illustrates the recognized NF model of the eNF predictor.

5.2.3. Worn gear monitoring

The prior crack testing is related to localized gear fault. Worn defect belongs to the category of distributed damage. At first, both gears in the gearbox are in healthy condition. The tests are conducted under load levels from 0.1 to 1.0 hp and motor speeds from 20 to 1400 rpm. To facilitate defect propagation, the tests are under overload conditions, whereas lubrication condition is set to an inappropriate condition in terms of lubricant type and level. The tests proceed until severe wear corresponding to scoring gear defect. In this test, all the related predictors are trained using the same old data set for pitting test as used in the aforementioned cracked gear testing. The three-steps-ahead ($s=3$) forecasting performance results are summarized in Table 5. It is clear that the developed eNF predictor performs more accurately in this test, in terms of efficiency (number of

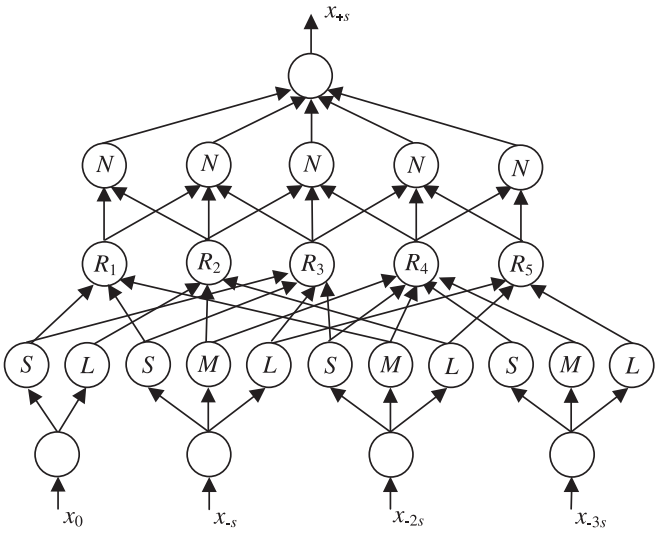


Fig. 11. The identified eNF model from the cracked gear testing.

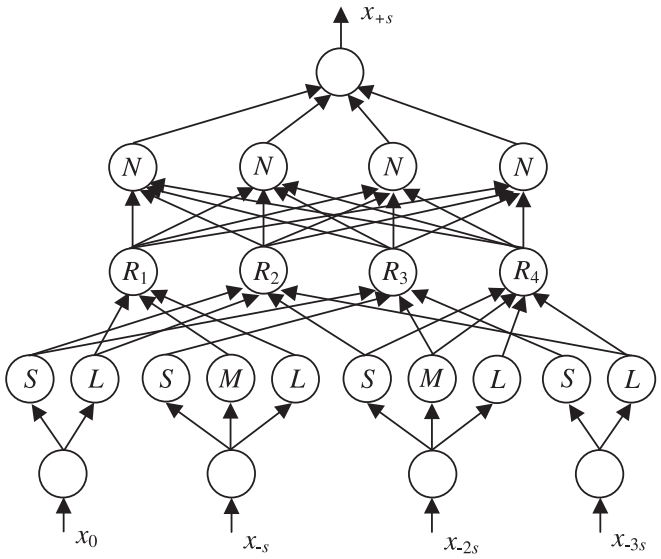


Fig. 13. The identified eNF model from the worn gear testing.

Table 5
Forecasting results (three-steps-ahead) of worn gear.

Forecasting schemes	Number of clusters	Average RMSE ($\times 10^{-2}$)s	Average training time per epoch (s)
TWNFI	8	2.98	0.14
eTS	6	2.33	0.12
eNF-0	4	1.67	0.15
eNF	4	1.26	0.12

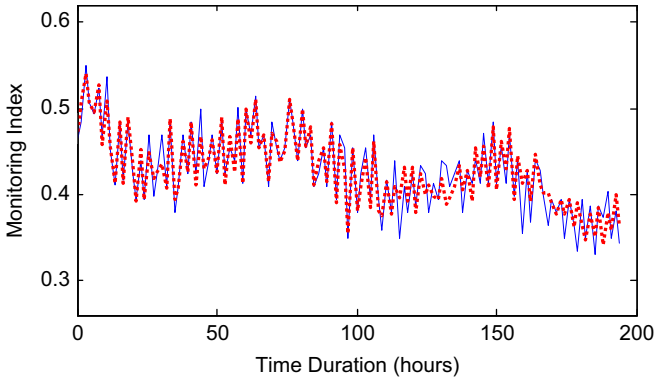


Fig. 12. Forecasting results (three-steps-ahead) of the healthy gear: real values (solid line) and the predicted values using the eNF predictor (dotted line).

clusters), accuracy, and convergence (speed of training). Although eNF-0 with the classical LSE training also generates 4 clusters, its accuracy and efficiency are lower than eNF predictor. Fig. 12 shows the forecasting results using the eNF predictor. It is seen that eNF can recognize and track the system's dynamic characteristics effectively. Fig. 13 illustrates the recognized eNF model.

As shown in Fig. 12, as the worn defect propagates, the monitoring indicator becomes smaller, even though the noise level becomes higher. This *healing* phenomenon is misleading in condition monitoring, which is associated with signal properties. From the point of view of signal properties, when a localized fault (e.g., crack or scoring) occurs, some high-amplitude pulses will be generated due to impacts, which are relatively easier for a signal processing technique to detect. As a distributed defect (e.g., wear or pitting) propagates, the overall energy of the fault will increase, but it often becomes more wideband in nature and difficult to detect in the presence of the other vibratory components of the machine. This example identifies a characteristic of currently used vibration-based signal processing techniques: it is usually easier to detect a distinct low-level narrowband tone than a high-level wideband signal in the presence of other signals or noises. In such case, signals from other sources such as acoustic and temperature signals should be used to comprehensively to diagnose distributed defects.

6. Conclusion

An evolving neuro-fuzzy predictor is developed in this paper to forecast the behavior of time-vary dynamic systems. An evolving clustering method is proposed for cluster generation and rule base modification. Based on the potential measurement and the mapping consistency, the contradictory fuzzy rules due to noise-affected clusters can be excluded effectively, whereas the resulting clusters are well-distributed over the input–output spaces. A novel enhanced LSE technique is proposed for online training of the linear parameters of the eNF predictor. The effectiveness of the developed eNF predictor is firstly evaluated by simulation using benchmark data sets, and then by real-world applications related to machinery condition monitoring and USD-CAD currency exchange rate forecasting. Test results have shown that the developed eNF predictor is an accurate and reliable forecasting tool. It can capture the system's dynamic behavior quickly and track the system's characteristics accurately. The online training technique can effectively improve the predictor's convergence and adaptive capability. It will be a promising forecasting tool for different real-world applications.

Acknowledgments

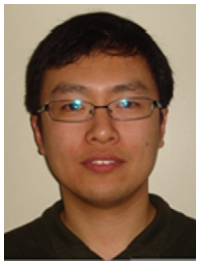
This project was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) and eMech Systems Inc.

References

[1] M. Pourahmadi, Foundations of Time Series Analysis and Prediction Theory, John Wiley & Sons Inc., 2001.
[2] S. Gupta, A. Ray, Real-time fatigue life estimation in mechanical structures, Meas. Sci. Technol. 18 (2007) 1947–1957.

[3] O. Dombayc, The prediction of heating energy consumption in a model house by using artificial neural networks in Denizli—Turkey, Adv. Eng. Software 41 (2010) 141–147.
[4] G. Vachtsevanos, F. Lewis, Intelligent Fault Diagnosis and Fault Prognosis for Engineering Systems, John Wiley & Sons, 2006.
[5] K. Aslantas, S. Tasgetiren, A study of spur gears pitting formation and life prediction, Wear 257 (2004) 1167–1175.
[6] L. Overbey, C. Olson, M. Todd, A parametric investigation of state-space-based prediction error methods with stochastic excitation for structural health monitoring, Smart Mater. Struct. 16 (2007) 1621–1638.
[7] S. BenTaieb, A. Sorjamaa, G. Bontempi, Multiple-output modeling for multi-step-ahead time series forecasting, Neurocomputing 73 (2010) 1950–1957.
[8] C. Li, H. Lee, Gear fatigue crack prognosis using embedded model, gear dynamic model and fracture mechanics, Mech. Syst. Signal Process. 9 (2005) 836–846.
[9] K. Lukoseviciute, M. Ragulskis, Evolutionary algorithms for the selection of time lags for time series forecasting by fuzzy inference systems, Neurocomputing 73 (2010) 2077–2088.
[10] J. Jang, ANFIS: adaptive-network-based fuzzy inference system, IEEE Trans. Syst. Man Cybern. 23 (1993) 665–685.
[11] W. Wang, An intelligent system for machinery condition monitoring, IEEE Trans. Fuzzy Syst. 16 (2008) 110–122.
[12] W. Wang, D. Kanneg, A smart predictor for material property testing, Smart Mater. Struct. 17 (2008) 17 065018.
[13] W. Wang, An enhanced diagnostic system for gear system monitoring, IEEE Trans. Syst. Man Cybern. B 34 (2008) 102–112.
[14] P. Angelov, Fuzzily connected multi-model systems evolving autonomously from data streams, IEEE Trans. Syst. Man Cybern. B 41 (2011) 898–910.
[15] P. Angelov, D. Filev, An approach to online identification of evolving Takagi–Sugeno models, IEEE Trans. Syst. Man Cybern. B 34 (2004) 484–498.
[16] E. Lughofer, P. Angelov, Handling drifts and shifts in on-line data streams with evolving fuzzy systems, Appl. Software Comput. 11 (2011) 2057–2068.
[17] Q. Song, N. Kasabov, TWNFI—a transductive neuro-fuzzy inference system with weighted data normalization for personalized modeling, Neural Networks 19 (2006) 1591–1596.
[18] J. Wang, C. Lee, Self-adaptive neuro-fuzzy inference systems for classification applications, IEEE Trans. Fuzzy Syst. 10 (2002) 790–802.
[19] W. Wang, J. Vrbanek, An evolving fuzzy predictor for industrial applications, IEEE Trans. Fuzzy Syst. 16 (2008) 1439–1449.
[20] R. Kumar, K. Izui, Y. Masataka, S. Nishiwaki, Multilevel redundancy allocation optimization using hierarchical genetic algorithm, IEEE Trans. Rel. 57 (2008) 650–661.
[21] M. Schliebs, D. Platel, S. Worner, N. Kasabov, Integrated feature and parameter optimization for evolving spiking neural networks: exploring heterogeneous probabilistic models, Neural Networks 22 (2009) 623–632.
[22] D. Li, W. Wang, An enhanced GA technique for system training and prognostics, J. Adv. Eng. Software 42 (2011) 452–462.
[23] M. Mackey, L. Glass, Oscillation and chaos in physiological control systems, Science 197 (1977) 287–289.
[24] SIDC, RWC Belgium World Data Center, Online Sunspot Data Archive, (2011).
[25] W. Antweiler, PACIFI Exchange Rate Service, 2011 〈http://fx.sauder.ubc.ca/data.html〉.

Wilson Wang received his M.Eng. in industrial engineering from the University of Toronto (Toronto, Ontario, Canada) in 1998 and Ph.D. in mechatronics engineering from the University of Waterloo (Waterloo, Ontario, Canada) in 2002, respectively. From 2002 to 2004, he was employed as a senior scientist at Mechworks Systems Inc. He joined the faculty of Lakehead University in 2004 and now he is an associate professor in the Department of Mechanical Engineering. His research interests include artificial intelligence, signal processing, machinery condition monitoring, time series forecasting, intelligent control, and bioinformatics.



De Z. (Derek) Li received his BASc in Electrical Engineering at Shandong University (Jinan, China) in 2008, and his M.Sc. in Control Engineering in 2010 at Lakehead University (Thunder Bay, Ontario, Canada). Currently he is a Ph.D. candidate in the Department of Mechanical and Mechatronics Engineering at the University of Waterloo (Waterloo, Ontario, Canada). His research interests include diagnosis and prognosis, artificial intelligence, and vibration control.

Joe Vrbanek received his BASc in Electrical Engineering with a minor in Economics in 2006, and M.Sc. in Control Engineering in 2008, both at Lakehead University (Thunder Bay, Ontario, Canada). Since 2008, he has been employed as a project manager at Hydro One Inc. (Toronto, Ontario, Canada). His research interests involve robotics and computational intelligence for classification and forecasting applications.