An evolving fuzzy neural predictor for multi-dimensional system state forecasting

De Z. Li\textsuperscript{a}, Wilson Wang\textsuperscript{b,\*}, Fathy Ismail\textsuperscript{a}

\textsuperscript{a} Department of Mechanical and Mechatronics Engineering, University of Waterloo, Waterloo, ON, Canada N2L 3G1
\textsuperscript{b} Department of Mechanical Engineering, Lakehead University, Thunder Bay, ON, Canada P7B 5E1

\section{Introduction}

Multi-dimensional system state forecasting is a complex and important research and development area, which aims to predict future states of a dynamic system based on past observations from multiple sources (e.g., sensors). The classical approaches for multi-dimensional data sets are generally based on analytical modeling, such as vector autoregressive (VAR) models, and vector autoregressive moving average (VARMA) models [1]. These classical analytical models can describe the underlying relationships among multi-dimensional data sets to extrapolate future states of a dynamic system, and have been used in some forecasting applications, such as the electricity load demand [2,3], and some economic indicators [4,5]. The VAR/VARMA models, however, can only predict the linear correlations among multi-dimensional data sets; they are unable to efficiently characterize nonlinear correlations (e.g., those related to impulses and transients) among multi-dimensional data sets. Comparing the VAR with the VARMA, the former estimates future states of a system based on its past multi-dimensional observations, while the latter deploys both past multi-dimensional observations and past multi-dimensional innovations for system state prediction. Thus VARMA could provide more comprehensive linear modeling than VAR. Since a long autoregressive (AR) process can be represented by a compact moving average (MA) process, the dimension of the parameter space may be further reduced by VARMA, especially for data sets involving long AR characteristics [6]. Accordingly, VARMA will be used in this work to filter out linear correlations among multi-dimensional data sets.

The alternative approach for multi-dimensional data set modeling is the use of soft-computing tools, such as neural networks (NNs) [7,8,17,18,26–29] and neural fuzzy (NF) systems [9,10]. An NF scheme is usually superior to NNs in mimicking human reasoning processes and extracting knowledge as interpretable IF–THEN rules. Although an NF scheme can track the nonlinear correlations among multi-dimensional data sets, the performance of NF in capturing linear correlations may not be efficient, because of its complex modeling nature. Moreover, the performance of an NF predictor with a fixed network architecture cannot be guaranteed when system properties vary significantly in applications (e.g., equipment just after repair and maintenance), and/or when new information is provided (e.g., from a new sensor) [11,12]. In recent years, more work has been focused on the use of evolving NF paradigms that can adaptively adjust their network structures in response to new system conditions (i.e., data sets).

\* Corresponding author. Tel.: +1 807 766 7174.
E-mail addresses: d458i@uwaterloo.ca (D.Z. Li), Wilson.Wang@Lakeheadu.ca (W. Wang), Fmassi1@uwaterloo.ca (F. Ismail).

http://dx.doi.org/10.1016/j.neucom.2014.05.014
0925-2312/© 2014 Elsevier B.V. All rights reserved.
Kasabov et al. successfully proposed evolving fuzzy NN models (EFuNN) [13–15] and dynamic evolving NF inference system (DENFIS) techniques [16] for different applications such as learning, knowledge acquisition, and time-series forecasting. These evolving paradigms employ clustering algorithms to adaptively tune model structure and parameters. Although they treat both linear correlations and nonlinear correlations in a data set with the fuzzy NN (nonlinear modeling), they may increase the computational burden, especially when they use nonlinear models to capture linear correlations from the data with large size and dimensions.

Another solution to model both linear and nonlinear characteristics of data sets for system state forecasting is the use of hybrid modeling. For example, Medeiros et al. proposed a neural coefficients of data sets for system state forecasting is the use of hybrid multi-dimensional data sets. Although the NNs can be pre-trained by the available multi-dimensional data sets to track system characteristics (mainly nonlinear correlations), they are inefficient in tracking linear correlations among multi-dimensional data sets. To properly tackle these modeling problems, an eFNN predictor is proposed in this section to provide a more efficient tool for prognosis of complex systems with multi-dimensional data sets involving both linear and nonlinear correlations.

### 2. The evolving fuzzy neural network

As stated in Section 1, although both VAR and VARMA models can catch the linear (but not nonlinear) correlations among multidimensional data sets, the VARMA is selected in this work for its more efficient generalization and compactness in modeling multi-dimensional data sets. Although the NNs can be pre-trained by the available multi-dimensional data sets to track system characteristics (mainly nonlinear correlations), they are inefficient in tracking linear correlations among multi-dimensional data sets. To properly tackle these modeling problems, an eFNN predictor is proposed in this section to provide a more efficient tool for prognosis of complex systems with multi-dimensional data sets involving both linear and nonlinear correlations.

#### 2.1. Architecture of the proposed eFNN predictor

Fig. 1 describes the network architecture of the proposed eFNN predictor. It is a six-layer feed-forward network. Layer 1 is the input layer. Each input \([x_{1,t}, x_{1,t-s}, \ldots, x_{1,t-(p-1)s}]\) represents a vector from the \(i\)th data set with time lags 0 to \(p-1\); \(s\) is the time-step; \(p\) is the dimension of the input data vector; \(l = 1, 2, \ldots, m\); and \(m\) is the dimension of the multiple data sets or the number of inputs to the eFNN predictor.

In forecasting applications, there exist both linear and nonlinear correlations between the target data set and these available data sets. Layer 2 performs VARMA filtering to model the linear correlation and nonlinear correlation in a data set: a compact VARMA filter to model linear property and an evolving fuzzy network to model nonlinear correlation. Based on this approach, the system structures can become more transparent, which can facilitate system training for optimization and error tracking.

The method’s novelty lies in the following aspects: (1) the developed eFNN predictor applies both linear and nonlinear modeling strategies to characterize properties of multi-dimensional data sets; (2) a novel cumulative clustering algorithm is proposed to evolve fuzzy reasoning rules for nonlinear modeling; and (3) the developed eFNN predictor is implemented for real-world applications such as the forecasting of currency exchange rates, as well as induction motor (IM) system state prognosis.

The remainder of this paper is organized as follows: The proposed eFNN predictor and the proposed adaptive clustering algorithm are discussed in Section 2. In Section 3, the effectiveness of the proposed eFNN predictor is examined by simulation tests; the new predictor is also implemented for IM system state prognosis. Some concluding remarks are summarized in Section 4.
where $\Theta = [\theta_{0,1} \theta_{0,2} \cdots \theta_{0,m}]$ are the linear AR parameters, 
$(k=1, 2, \ldots, p)$, and $\Lambda = [\lambda_{1,1} \lambda_{1,2} \cdots \lambda_{1,m}]$ are the linear MA parameters; 
$(l=1, 2, \ldots, q)$. $Y_t$ is the predicted value of the $i$th dimensional data set using VARMA.

The linear filtering output $Y_t$ is forwarded to the output node in Layer 6. $\phi_{ij}$ is the linear filtering error of the $i$th dimensional data set at time instance $j$. To conduct an $s$-step-ahead forecasting, $\phi_{ij}$ can be determined as $\phi_{ij} = Y_{0} - Y_t$, where $Y_0$ and $Y_t$ are the desired and estimated values of the VARMA filter in the $i$th dimensional data set at time instant $j$, respectively.

To simplify representation, the linear estimation (or filtering) errors $\{\phi_{1,1}, \phi_{2,1}, \cdots, \phi_{m,1}\}$ are represented as $\{\phi_{1}, \phi_{2}, \cdots, \phi_{m}\}$, which are the inputs in Layer 3.

Layer 4 is the fuzzy rule layer. Gaussian functions are selected as membership functions (MFs). $B_{ji}$, to formulate the fuzzy operation. Given the input $\Phi = [\phi_{1}, \phi_{2}, \cdots, \phi_{m}]$, the firing strengths $\eta_j$ can be derived by using the fuzzy product $T$-norm

$$
\eta_j = \exp \left(-\frac{1}{2} \frac{||\Phi - \mu_j||^2}{\sigma_j^2} \right)
$$

$$
= \exp \left(-\frac{1}{2} \frac{\phi_1^2 + \phi_2^2 + \cdots + \phi_m^2}{\sigma_j^2} \right) = \prod_{i=1}^{m} B_{ji}(\phi_i)
$$

\hspace{1cm} (2)

and

$$
B_{ji}(\phi_i) = \exp \left(-\frac{1}{2} \frac{(\phi_i - \mu_j)^2}{\sigma_j^2} \right)
$$

\hspace{1cm} (3)

where $\mu_j = [\mu_{j,1}, \mu_{j,2}, \cdots, \mu_{j,m}]$ is the center of the $j$th cluster, which will be derived using a clustering algorithm. The clustering technique will be introduced in Section 2.2. $\mu_{j,1}, \mu_{j,2}, \cdots, \mu_{j,m}$ and $\sigma_j$ are the respective center and spread of the Gaussian MF $B_{ji}$. $n$ is the number of nodes for each input $\phi_i$. Layer 5 is the rule layer. Each node in this layer is formulated by $L_j = a_{j,1}x_1 + a_{j,2}x_2 + \cdots + a_{j,m}x_m + b_j$

\hspace{1cm} (4)

where $L_j$ denotes a first order TS model, in which $a_{j,i} (j=1, 2, \ldots, n; i=1, 2, \ldots, m)$ are the linear parameters, and $b_j$ is the bias in $L_j$.

Layer 6 is the output layer. The eFNN output $Y$ is formulated as

$$
Y = Y_t + \sum_{j=1}^{n} \frac{1}{\eta_j} L_j
$$

\hspace{1cm} (5)

The fuzzy rules in Layers 4–5 are generated by the use of an evolving clustering paradigm that will be discussed in Section 2.2.

2.2. The adaptive clustering algorithm

A cumulative evolving clustering (CEC) algorithm is proposed in this work, to adaptively evolve the fuzzy reasoning rules (clusters) represented in Layers 4–5 in Fig. 1. The inputs to the evolving fuzzy (EF) network are the linear estimation error vectors $\Phi = [\phi_1, \phi_2, \cdots, \phi_m]$, and the $j$th fuzzy rule can be formulated as $R_j: IF \ (\phi_1 \ is \ B_{1,j}) \ and \ (\phi_2 \ is \ B_{2,j}) \ and, \ \ldots, \ and \ (\phi_m \ is \ B_{m,j}),$

then $Z \ is \ L_t$.

\hspace{1cm} (6)

where $j = 1, 2, \ldots, n$ is the number of fuzzy rules generated; $Z$ is the clustering index; $h = 1, 2, \ldots, r$; and $r$ is the number of first order TS models ($r \leq n$). The generated $j$th cluster is an $m$-dimensional cluster with center $C_j = [C_{j,1}, C_{j,2}, \cdots, C_{j,m}]$, and radius $R_j$.

Fig. 2 schematically illustrates the clustering process of the CEC algorithm. The normalized Euclidean distance between the new input $\Phi$ and the center of the $j$th cluster, $C_j$, is defined as

$$
d_j = \frac{1}{\sqrt{m}} ||\Phi - C_j||
$$

\hspace{1cm} (7)

Step 1. Initialization: When the first input $\Phi$ (i.e., linear estimation error vector) is formulated after VARMA filtering, it becomes the center of the first cluster, $C_1$. The initial radius of this cluster is set as $R_1$ ($R_0 = 0.01$, in this case). The upper boundary of the radius is denoted by $R_0$. If $R_1$ is small, more clusters will be generated, and vice versa.

Step 2. Cluster formulation: If a new input $\Phi$ falls in more than one cluster, only the center and radius of the closest cluster will be updated using the following rules:

(a) If $d_j < R_j$ (e.g., state $A$ in Fig. 2), the center of the cluster is updated as

$$
C_{j,\text{new}} = (C_{j,\text{old}} \times N_j + \Phi)/(N_j + 1)
$$

\hspace{1cm} (8)

where $N_j$ is the number of input values in cluster $j$.

(b) If $R_j < d_j < R_0$ (e.g., state $B$ in Fig. 2), the center and radius of this cluster remain unchanged.

(c) Otherwise, If $R_j < d_j < (2R_0 - 2R_k)$ (e.g., state $D$ in Fig. 2), the center of the cluster remains unchanged, but the radius $R_j$ is updated as $R_j = (1/2)d_j + R_k$.

(d) If $R_j > (2R_0 - 2R_k) < d_j$ (e.g., state $E$ in Fig. 2), a new cluster is created. The new input value $\Phi$ becomes the center of the new cluster and the radius is initialized as $R_0$.

(e) If $2R_0 - 2R_k < R_j < d_j$, a new cluster is created with the same setting as in (d).

In general, the data with $d_j < R_j$ will have higher MF degree, and will be used to determine the cluster center. The data with $R_j < d_j < (2R_0 - 2R_k)$ represent the potential spread of the cluster, and will be used to update the radius $R_j$ of the cluster. If the input data satisfies $d_j < R_j < R_k$, the parameters of the cluster remain unchanged.

Step 3. Structure recognition: The center of the $j$th cluster $C_j$ will be the center of the $j$th firing strength (i.e., $\eta_j$ in Eq. (2)). Assume that the center $C_j$ corresponds to the MF degree 100% (i.e., $\eta_j = 1$), and the input values $\Phi_k$ with $d_j = R_j$ are assigned a MF degree of 1% (i.e., $\eta_j = 0.01$).

Substituting $\eta_j = 0.01$ in Eq. (2), yields

$$
\exp \left(-\frac{1}{2} \frac{||\Phi - \mu_j||^2}{\sigma_j^2} \right) = 0.01
$$

\hspace{1cm} (9)
Then inserting Eq. (7) into Eq. (9) gives the following equation:

$$\exp\left(-\frac{1}{2}\frac{m R^2}{\sigma_j^2}\right) = 0.01$$  \hspace{1cm} (10)

The spread $\sigma_j$ can be derived by rearranging Eq. (10)

$$\sigma_j = \sqrt{\frac{m R_j}{2 \ln(0.01)}}$$  \hspace{1cm} (11)

When an input value with $d_j \leq R_j$, its MF degree can be derived as

$$\eta_j = \exp\left(-\frac{1}{2}\frac{|\theta - \mu_j|^2}{\sigma_j^2}\right) = \exp\left(-\frac{1}{2}\frac{d_j^2}{R_j^2}\right) \ln(0.01)$$  \hspace{1cm} (12)

Since $(d_j/R_j)^2 \leq 1$ and $\ln(0.01)$ is a negative value, the MF degree $\eta_j = \exp\left((d_j/R_j)^2 \ln(0.01)\right) \geq 0.01$. When $d_j$ takes the extremely small value (i.e., $d_j = 0$), the MF degree $\eta_j = 1$. Therefore the MF degree $\eta_j$ will be within the range of $[0, 1]$.

When projecting the m-dimensional data sets, the corresponding MF center and radius will be center $\mu_j$ and spread $\sigma_j$ of $R_j$ in Eq. (3), respectively. The proposed CEC is a constrained evolving algorithm, which is dependent on two factors: the input $\Phi$, and the upper boundary $R_j$. If $R_j$ remains constant (a general case), the clusters will be evolved based only on input information $\Phi$.

The proposed CEC updates the center of a cluster by considering the information of both previous cluster center position (i.e., Eq. (8)) and the newest input sample. Thus the updated cluster center of the CEC is less sensitive to outliers. The CEC adjusts the cluster center using an evolving mechanism, $R_j^t = (1/2) + ((M - N_j)/(2M))R_j$. These new clusters (containing only a few samples) will have more opportunities to update their centers using new accommodated samples, so as to optimize cluster center position. The centers of the clusters with many samples will be less sensitive to new accommodated samples, especially those new samples which belong to this cluster, but are far away from the cluster center. Therefore the cluster center can be less affected by outliers.

2.3. Training strategy

The parameters of the developed eFNN predictor will be optimized by appropriate training as illustrated in Fig. 3. To catch linear correlations of the m-dimensional data sets, the parameters in the VARMA filter are optimized online by the use of the recursive least square estimate (RLSE) suggested by the authors [21].

A hybrid training strategy will be used to optimize parameters in the EF network (Layers 3–5 in Fig. 1). The gradient descent (GD) algorithm is employed to update the nonlinear parameters in nodes $B_{lj}$ in Layer 4, whereas the RLSE is utilized to adaptively tune the linear parameters in $L_i$ in Eq. (4). According to our previous research in system training [20], a hybrid training strategy can reduce the search dimension when compared with a single training method (e.g., the GD), prevent becoming trapped in local optima, and improve convergence of the training process.

Specific training processes are summarized as follows:

1. The initial values of parameters in nodes $L_i$ ($j=1, 2, ..., n$) are initialized over the interval $[0, 1]$.
2. The parameters $\theta_{kj}$ and $\lambda_{ij}$ ($k=1, 2, ..., p; l=1, 2, ..., q; i, j=1, 2, ..., m$) in the VARMA filter are optimized online by using the RLSE.
3. After training of the VARMA filter parameters, the nonlinear parameters in nodes $B_{lj}$ ($j=1, 2, ..., n$; $i=1, 2, ..., m$) are optimized by using a GD algorithm, and linear parameters in $L_i$ are updated by RLSE adaptively.

In the training process, only one training epoch is needed to update the VARMA filter parameters. After the linear correlation information is filtered out, the estimation error data set $[\varphi_1, \varphi_2, ..., \varphi_m]$ will retain less regulated information. This means that fewer clusters can be formulated, which can simplify the eFNN predictor structure and speed up training convergence, as discussed in Section 3.

3. Performance evaluation and applications

3.1. Overview

The effectiveness of the proposed eFNN predictor is verified in this section, first by developing a forecasting example of the multi-dimensional financial data set, and then being implemented for IM system state prognosis. To simplify the discussion, the developed eFNN predictor using the proposed CEC algorithm is designated as eFNN-CEC.

To make a comparison, the related predictors based on an enhanced fuzzy filtered neural network (EFFNN) [20], an eNF scheme [21], and the DENFIS [13] are employed for testing. The EFFNN predictor is a four-layer feed-forward NN with the same number of nodes from Layers 1 to 3 (i.e., 20–20–20–1). The eNF predictor is an evolving NN with three input nodes. The DENFIS is a data-driven NN using a clustering technique.

To evaluate the effectiveness of the proposed evolving CEC algorithm in the eFNN predictor, an evolving clustering method (ECM) suggested in [13] is implemented in the eFNN predictor to replace the CEC algorithm, designated here as eFNN-ECM. That is, the only difference between the eFNN-CEC and the eFNN-ECM is associated with evolving clustering algorithms. The maximum number of training epochs of the predictors (i.e., EFFNN, eNF, eFNN-ECM, and eFNN-CEC) is set at 1000.

3.2. Simulation example: exchange rate forecasting

In this test, a six-dimensional currency exchange rate data set is used to examine the performance of the proposed eFNN-CEC predictor. The data set consists of daily exchange rates of Canadian dollar versus US dollar (the first dimensional data set), European euro versus US dollar (the second dimensional data set), British pound versus US dollar (the third dimensional data set), Australian

Fig. 3. Flowchart of the hybrid training process of the developed eFNN predictor.
dollar versus US dollar (the fourth dimensional data set), Hong Kong dollar versus US dollar (the fifth dimensional data set) and New Zealand dollar versus US dollar (the sixth dimensional data set). All of them were collected simultaneously over the period from January 1, 2010 to May 31, 2012 [22]. The tests are performed on each dimensional data set. The first one-third of the data set in each dimension is used for training, and the remainder is used for testing.

![Fig. 4. Comparison of three-step-ahead forecasting results of daily Canadian dollar/US dollar exchange rate data. The blue solid line is the real data to estimate; the red dotted line is the forecasting results using different predictors: (a) EFFNN; (b) eNF; (c) DENFIS; (d) eFNN-ECM; and (e) eFNN-CEC. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)](image)

![Fig. 5. Comparison of three-step-ahead forecasting results of daily European euro/US dollar exchange rate data. The blue solid line is the real data to estimate; the red dotted line is the forecasting results using different predictors: (a) EFFNN; (b) eNF; (c) DENFIS; (d) eFNN-ECM; and (e) eFNN-CEC. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)](image)
Figs. 4–9 illustrate the three-step-ahead forecasting performance of these six data sets using the related predictors, and the related results are summarized in Tables 1–6. It is seen that the eNF predictor is superior to the EFFNN predictor in terms of both forecasting accuracy and the running time, because of its evolving reasoning mechanism. The DENFIS achieves less prediction error than the eNF as indicated in Tables 1, 3, 4 and 6, but with more clusters generated and hence more running time, because more fuzzy rules are used to capture the data characteristics. In Tables 2 and 5, the eNF outperforms the DENFIS in terms of number
clusters generated, prediction errors and running time, because of its more advanced evolving mechanism. The eFNN related predictors (i.e., eFNN-CEC and eFNN-ECM) yield less forecasting error than those based on the DENFIS, the eNF and the EFFNN, because eFNN undertakes more efficient linear and nonlinear correlation modeling (e.g., eFNN-ECM generates 75% and 67% less errors than the eNF and the DENFIS, respectively, as demonstrated in Table 1). The error in percentage is calculated as $E(\%) = 100 \times (E_A - E_B) / E_B$; $E_A$ and $E_B$ represent the MSEs of predictors $A$ and $B$, respectively. The developed CEC evolving method in eFNN-CEC is more efficient than the classical ECM in eFNN-ECM (e.g., eFNN-CEC generates 42% less error than eFNN-ECM in Table 4).
On the other hand, eFNN-CEC formulates the fewest clusters in these tests because of its linear filtering operation and effective clustering algorithms, which takes less running time. Therefore, the eFNN-CEC predictor outperforms other predictors in capturing and tracking the dynamic behaviors of the underlying systems in this test.

### 3.3. Application example: induction motor system state prognosis

The developed eFNN-CEC predictor is implemented for IM system state prognosis. System state prognosis is an important strategy for equipment health condition monitoring [23]. An efficient predictor is very helpful in estimating an IM’s dynamic characteristics for system state prognosis and performance control. Fig. 10 shows the experimental setup used in this test. The speed of the tested 3-phase IM is controlled by a speed controller (VFD-B from Delta Electronics) with output frequency 0.1–400 Hz. A magnetic particle clutch (PHC-50 from Placid Industries) is used as a dynamometer for external loading. Its torque range is from 1 to 41 Nm. The IM used for tests is made by Marathon Electric. The gearbox (Boston Gear 800) is used to adjust the speed ratio of the dynamometer. A Quanser Q4 data acquisition board is used for data acquisition.

This test is to forecast future states of phase current signals that will be used for IM health condition monitoring. During the test, phase current signals are collected at a sampling frequency of 10 kHz. Two current signals are used to form a two-dimensional data set for IM system state forecasting in this case. The first dimensional data set is a current residual signal from phase 1 by filtering out supply frequency components, and the second dimensional data set is a current signal from phase 2. The first 300 data are used for training, and the remaining 600 data are used for testing.

Figs. 11 and 12 respectively show the respective four-step-ahead forecasting results of the signal residual and the current signal using the related predictors, and the results are summarized in Tables 7 and 8. The unit of the induction motor current signal is in Amperes (A). After calculating the MSE of the predicted values, the unit becomes A² that is used in Tables 7 and 8. It can be seen that the eNF outperforms the EFFNN, with 29% less error in Table 7. The DENFIS generates more clusters and would take longer running time than the eNF, but the DENFIS is more accurate than eNF because more fuzzy rules are used to model the data characteristics. The eFNN predictors outperform predictors based on the EFFNN, the eNF and the DENFIS because the eFNN can employ both linear and nonlinear modeling mechanisms. Moreover, the eFNN-CEC predictor creates the fewest clusters (only three in this case) compared to eFNN-ECM (four–five clusters), eNF (six–eight clusters) and DENFIS (12–13 clusters). From Tables 7 and 8,

<table>
<thead>
<tr>
<th>Forecasting schemes</th>
<th>No. of clusters</th>
<th>MSE (10⁻⁴)</th>
<th>Running time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EFFNN</td>
<td>20</td>
<td>6.072</td>
<td>27.211</td>
</tr>
<tr>
<td>eNF</td>
<td>6</td>
<td>3.723</td>
<td>22.524</td>
</tr>
<tr>
<td>DENFIS</td>
<td>10</td>
<td>2.797</td>
<td>25.979</td>
</tr>
<tr>
<td>eFNN-ECM</td>
<td>3</td>
<td>0.918</td>
<td>17.872</td>
</tr>
<tr>
<td>eFNN-CEC</td>
<td>2</td>
<td>0.421</td>
<td>16.013</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Forecasting schemes</th>
<th>No. of clusters</th>
<th>MSE (10⁻⁴)</th>
<th>Running time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EFFNN</td>
<td>20</td>
<td>15.000</td>
<td>27.422</td>
</tr>
<tr>
<td>eNF</td>
<td>8</td>
<td>4.579</td>
<td>22.492</td>
</tr>
<tr>
<td>DENFIS</td>
<td>10</td>
<td>8.843</td>
<td>26.634</td>
</tr>
<tr>
<td>eFNN-ECM</td>
<td>3</td>
<td>3.990</td>
<td>21.261</td>
</tr>
<tr>
<td>eFNN-CEC</td>
<td>2</td>
<td>2.689</td>
<td>18.813</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Forecasting schemes</th>
<th>No. of clusters</th>
<th>MSE (10⁻⁴)</th>
<th>Running time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EFFNN</td>
<td>20</td>
<td>9.281</td>
<td>26.113</td>
</tr>
<tr>
<td>eNF</td>
<td>7</td>
<td>6.448</td>
<td>24.804</td>
</tr>
<tr>
<td>DENFIS</td>
<td>10</td>
<td>3.502</td>
<td>25.462</td>
</tr>
<tr>
<td>eFNN-ECM</td>
<td>3</td>
<td>2.201</td>
<td>16.973</td>
</tr>
<tr>
<td>eFNN-CEC</td>
<td>2</td>
<td>1.279</td>
<td>16.109</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Forecasting schemes</th>
<th>No. of clusters</th>
<th>MSE (10⁻⁴)</th>
<th>Running time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EFFNN</td>
<td>20</td>
<td>4.436</td>
<td>27.594</td>
</tr>
<tr>
<td>eNF</td>
<td>8</td>
<td>0.926</td>
<td>25.481</td>
</tr>
<tr>
<td>DENFIS</td>
<td>10</td>
<td>2.321</td>
<td>26.687</td>
</tr>
<tr>
<td>eFNN-ECM</td>
<td>4</td>
<td>0.883</td>
<td>17.521</td>
</tr>
<tr>
<td>eFNN-CEC</td>
<td>2</td>
<td>0.738</td>
<td>15.506</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Forecasting schemes</th>
<th>No. of clusters</th>
<th>MSE (10⁻⁴)</th>
<th>Running time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EFFNN</td>
<td>20</td>
<td>8.463</td>
<td>26.524</td>
</tr>
<tr>
<td>eNF</td>
<td>7</td>
<td>3.951</td>
<td>24.479</td>
</tr>
<tr>
<td>DENFIS</td>
<td>10</td>
<td>3.332</td>
<td>24.941</td>
</tr>
<tr>
<td>eFNN-ECM</td>
<td>4</td>
<td>1.411</td>
<td>17.893</td>
</tr>
<tr>
<td>eFNN-CEC</td>
<td>2</td>
<td>1.141</td>
<td>16.416</td>
</tr>
</tbody>
</table>
it is clear that the proposed eFNN-CEC predictor provides the highest forecasting accuracy (in terms of MSE) when compared with eFNN-ECM, eNF, DENFIS and EFFNN predictors. It can be seen from Table 8 that the eFNN-CEC generates 43% less error than the second best predictor, eFNN-ECM. The eFNN-CEC predictor can catch the dynamic behavior of the tested IM system quickly and accurately.

Fig. 11. Comparison of four-step-ahead forecasting results of the signal residual of an IM. The blue solid line is the real data to estimate; the red dotted line is the forecasting results using different predictors: (a) EFFNN; (b) eNF; (c) DENFIS; (d) eFNN-ECM; and (e) eFNN-CEC. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Fig. 12. Comparison of four-step-ahead forecasting results of the current signal of an IM. The blue solid line is the real data to estimate; the red dotted line is the forecasting results using different predictors: (a) EFFNN; (b) eNF; (c) DENFIS; (d) eFNN-ECM; and (e) eFNN-CEC. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

The IM data is a two dimensional data set. If the proposed eFNN-CEC technique is used, the residual IM data after VARMA filtering can be represented as $\Phi = [\phi_1, \phi_2]$ in Eq. (2), which is then fed to EF network for nonlinear modeling. The proposed CEC technique is used to generate clusters from the input patterns $\Phi$, so as to adaptively construct the structure of EF network as well as
Gaussian MFs $B_{ij}$ in Eq. (3). The distributions of input patterns $\phi$, and the corresponding trained Gaussian MFs are shown in Figs. 13 and 14, respectively. The parameters $\mu_j, 1, \mu_j, 2$ and $s_j$ of the Gaussian MFs in Eq. (3) corresponding to Figs. 13 and 14, are given in Tables 9 and 10, respectively. From Figs. 13 and 14, it is seen that the derived three Gaussian MFs can capture the distribution of the input patterns effectively. Compared to other predictors with more generated clusters (or Gaussian MFs), the proposed eFNN-CEC can conduct accurate prediction with less running time as demonstrated in Tables 7 and 8.

### Table 7
Forecasting results (four-step-ahead) of the signal residual of an IM (from phase 1).

<table>
<thead>
<tr>
<th>Forecasting schemes</th>
<th>No. of clusters</th>
<th>MSE ($A^2$)</th>
<th>Running time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EFFNN</td>
<td>20</td>
<td>0.021</td>
<td>41.593</td>
</tr>
<tr>
<td>eNF</td>
<td>8</td>
<td>0.015</td>
<td>38.988</td>
</tr>
<tr>
<td>DENFIS</td>
<td>13</td>
<td>0.011</td>
<td>40.124</td>
</tr>
<tr>
<td>eFNN-ECM</td>
<td>5</td>
<td>0.008</td>
<td>25.553</td>
</tr>
<tr>
<td>eFNN-CEC</td>
<td>3</td>
<td>0.002</td>
<td>20.704</td>
</tr>
</tbody>
</table>

### Table 8
Forecasting results (four-step-ahead) of the current signal of an IM (from phase 2).

<table>
<thead>
<tr>
<th>Forecasting schemes</th>
<th>No. of clusters</th>
<th>MSE ($A^2$)</th>
<th>Running time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EFFNN</td>
<td>20</td>
<td>0.018</td>
<td>39.051</td>
</tr>
<tr>
<td>eNF</td>
<td>6</td>
<td>0.037</td>
<td>34.129</td>
</tr>
<tr>
<td>DENFIS</td>
<td>12</td>
<td>0.008</td>
<td>37.478</td>
</tr>
<tr>
<td>eFNN-ECM</td>
<td>4</td>
<td>0.007</td>
<td>20.732</td>
</tr>
<tr>
<td>eFNN-CEC</td>
<td>3</td>
<td>0.004</td>
<td>16.046</td>
</tr>
</tbody>
</table>

**Fig. 13.** The distribution of (a) input patterns $\phi_1$ and (b) input patterns $\phi_2$, associated with the corresponding Gaussian MFs, for the signal residual of an IM. The blue solid line represents the distribution of the input patterns; the red dotted line represents the Gaussian MFs. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

**Fig. 14.** The distribution of (a) input patterns $\phi_1$ and (b) input patterns $\phi_2$, associated with the corresponding Gaussian MFs, for the current signal of an IM. The blue solid line represents the distribution of the input patterns; the red dotted line represents the Gaussian MFs. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

### Table 9
The parameters of the trained Gaussian MFs for the signal residual of an IM.

<table>
<thead>
<tr>
<th>Parameters $\mu_{ji}$ (Fig. 13a)</th>
<th>$\mu_{ji}$ (Fig. 13b)</th>
<th>$\sigma_j$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster 1 ($j=1$)</td>
<td>0.254</td>
<td>0.048</td>
</tr>
<tr>
<td>Cluster 2 ($j=2$)</td>
<td>0.734</td>
<td>0.082</td>
</tr>
<tr>
<td>Cluster 3 ($j=3$)</td>
<td>-0.303</td>
<td>0.036</td>
</tr>
</tbody>
</table>

### Table 10
The parameters of the trained Gaussian MFs for the current signal of an IM.

<table>
<thead>
<tr>
<th>Parameters $\mu_{ji}$ (Fig. 14a)</th>
<th>$\mu_{ji}$ (Fig. 14b)</th>
<th>$\sigma_j$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster 1 ($j=1$)</td>
<td>0.288</td>
<td>-0.028</td>
</tr>
<tr>
<td>Cluster 2 ($j=2$)</td>
<td>-0.333</td>
<td>-0.001</td>
</tr>
<tr>
<td>Cluster 3 ($j=3$)</td>
<td>-0.077</td>
<td>0.211</td>
</tr>
</tbody>
</table>

### 4. Conclusion

An evolving fuzzy neural network (eFNN) predictor has been developed in this work for multi-dimensional system state forecasting. It integrates the advantages of both the VARMA filter and nonlinear network modeling approaches in dealing with multi-dimensional data sets. A novel evolving clustering algorithm, CEC, is proposed to adaptively generate fuzzy reasoning clusters and adjust the eFNN network structure. The effectiveness of the proposed eFNN predictor and the new clustering algorithm is verified by simulation using a multidimensional financial data set. The new predictor has also been implemented for the induction motor (IM) system state prognosis. Test results have showed that the developed eFNN predictor is an accurate forecasting tool, and can capture the dynamic behavior of the tested system quickly and accurately. The CEC is also an effective evolving technique for network structure formulation and improvement.

**Acknowledgment**

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


De Z. Li received his B.Sc. degree in Electrical Engineering from the Shandong University, Jinan, China, in 2008, and M.Sc. degree in Control Engineering from the Lakehead University, Thunder Bay, ON, Canada in 2010. From 2010 to 2011, he was a Research Associate at Lakehead University. He is currently a Ph.D. candidate with the Department of Mechanical & Mechatronics Engineering at University of Waterloo. His research interests include signal processing, machinery condition monitoring, mechatronic systems, linear/nonlinear system control and artificial intelligence.

Wilson Wang received his M.Eng. in Industrial Engineering from the University of Toronto, Toronto, ON, Canada, in 1998 and the Ph.D. in Mechatronics Engineering from the University of Waterloo, Waterloo, ON, Canada, in 2002. From 2002 to 2004, he was a Senior Scientist with Mechworks Systems Inc. He joined the faculty of Lakehead University, Thunder Bay, ON, Canada, in 2004, where he is currently a Professor with the Department of Mechanical Engineering. His research interests include signal processing, artificial intelligence, machinery condition monitoring, intelligent control and mechatronics.

Fathy Ismail received the B.Sc. and M.Sc. degrees in Mechanical and Production Engineering, in 1970 and 1974, respectively, from the Alexandria University, Egypt, and the Ph.D. degree from the McMaster University, Hamilton, Ontario, Canada, in 1983. He joined the University of Waterloo, Waterloo, Ontario, Canada, in 1983, and is currently a Professor in the Department of Mechanical and Mechatronics Engineering. He has served as the Chair of the Department and the Associate Dean of the Faculty of Engineering for Graduate Studies. His research interests include machining dynamics, high-speed machining, modeling structures from modal analysis testing, and machinery health condition monitoring and diagnosis.