A New Data-Driven Neural Fuzzy System with Collaborative Fuzzy Clustering Mechanism

M. Prasad[a], Y. Y. Lin[b], C. T. Lin[c, *], M. J. Er[d], O. K. Prasad[e]

[a] Department of Computer Science, [b, c] Brain Research Center, [c] Department of Electrical Engineering,

[e] EECS International Graduate Program, National Chiao Tung University, Hsinchu, Taiwan

[d] School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore

[a] mukeshnctu.cs99g@nctu.edu.tw, [b] oliver.yylin@gmail.com, [c,] ctlin@mail.nctu.edu.tw,*

[d] EMJER@ntu.edu.sg, [e] omg.eo02g@nctu.edu.tw

Abstract

In this paper, a novel fuzzy rule transfer mechanism for self-constructing neural fuzzy inference networks is being proposed. The features of the proposed method, termed data-driven neural fuzzy system with collaborative fuzzy clustering mechanism (DDNFS-CFCM) are; (1) Fuzzy rules are generated facilely by fuzzy c-means (FCM) and then adapted by the preprocessed collaborative fuzzy clustering (PCFC) technique, and (2) Structure and parameter learning are performed simultaneously without selecting the initial parameters. The DDNFS-CFCM can be applied to deal with big data problems by the virtue of the PCFC technique, which is capable of dealing with immense datasets while preserving the privacy and security of datasets. Initially, the entire dataset is organized into two individual datasets for the PCFC procedure, where each of the dataset is clustered separately. The knowledge of prototype variables (cluster centers) and the matrix of just one halve of the dataset through collaborative technique are deployed. The DDNFS-CFCM is able to achieve consistency in the presence of collective knowledge of the PCFC and boost the system modeling process by parameter learning ability of the self-constructing neural fuzzy inference networks (SONFIN). The proposed method outperforms other existing methods for time series prediction problems.

Keywords: Neural networks, Fuzzy system, Big-data, Privacy and security, Collaborative technique, On-line learning system, Time series prediction.

1. INTRODUCTION

Neural networks and fuzzy systems [1] are two important technologies which play a pivotal role towards realization of machine learning and artificial intelligence [2]. The integration of fuzzy inference systems (FISs) and artificial neural networks (ANNs) has been widely pursued by many researchers due to the requisite of adaptive intelligent systems for solving real-world problems. The integrated technology, called neural fuzzy technique, has been applied frequently in many disciplines related to engineering. Consequently, many researchers have focused on system modeling by using neural fuzzy techniques [3-5], because it possesses the advantages of both neural networks and fuzzy systems. Moreover, structure identification and parameter learning of neural fuzzy networks help prevailing over the incapability of fuzzy systems with parameter learning and neural networks unable to do interpretation of human-like intelligence. In neural fuzzy systems, several data-driven strategies to generate appropriate numbers of fuzzy rules have been introduced [6-8].

Rong and Sundarajan [9] proposed a sequential adaptive fuzzy inference system (SAFIS) based on the functional equivalence between a radial basis function network and a fuzzy inference system (FIS). In this method, if there is no admission to the new fuzzy rule by input data, then only the parameters of the nearest rule are updated by using an extended Kalman filter (EKF) scheme. Dovzan and Skrjan [10] proposed an on-line TSK-type fuzzy model, which can be used for modeling control system or robotics by combination of a recursive fuzzy c-means and least squares. This method needs more computational cost than the SAFIS because of the fuzzy covariance matrix. However, the memory requirements are stationary due to the inelastic number of clusters. Wang and Lee [11] proposed a self-adaptive neuro-fuzzy inference system (SANFIS), which is adequate for self-adapting and self-organizing its domestic structure to obtain an economical rule base for illustrating the internal structure from input training dataset of the system. An online sliding-window-based self-organizing fuzzy neural network (SOFNN) was proposed by Leng and Prasad [12], which is suitable for machine learning and also it is applicable for cognitive reasoning in smart home environment. Er and Wu [13] proposed a learning algorithm for dynamic fuzzy neural networks (DFNN) based on extended radial basis function (RBF) neural networks. The features of DFNN approach evolve around free parameters that can be adjusted and structure learning mechanism associated with self-adaptive operation through a pruning technique.

Wang [14] proposed a generalized-ellipsoidal-basis-function-based online self-constructing fuzzy neural network (GEBF-OSFNN), which enlarges the ellipsoidal basis function (EBF)-based fuzzy neural networks (FNNs) by allowing the input variables to be modeled by dissymmetrical Gaussian functions (DGFs). Han [15] proposed a novel growing-and-pruning (GP) approach, which improves the formation of fuzzy neural networks (FNNs). The GP-FNN is based on RBFN, where neither the parameters nor the numbers of neurons in the hidden layer requires, all values are allocated during the learning process. Reinforcement evolutionary learning algorithm (REL) was proposed by Lin and Chen [16] for self-evolving neural fuzzy inference networks (SENFIN). The proposed REL consists of parameter learning and structure learning which are used to adjust the parameters of the SENFIN and determine the number of fuzzy rules. The merits of the SENFIN-REL technique include that it can dynamically design the structure of SENFIN and adjust free parameters of SENFIN whose consequent part is a nonlinear combination of input variables. Malek [17] proposed three new hybrid learning algorithms for Takagi-Sugeno-Kang fuzzy systems by using three kinds of manners, including the K-nearest neighbor, mean-shift procedure and space partitioning, which are more effective in terms of accuracy and requires fewer rules because of the simplicity of the algorithms with lower computational cost and approximate nonlinear functions. It has been shown that fixing the variance value for the Gaussian fuzzy sets reduces the number of parameters and there is no need for parameter tuning.

The existing fuzzy neural networks (FNNs) have two factions. The first faction is fuzzy systems with self-tuning ability but it requires initialization of the number of fuzzy rules. The second faction of neural fuzzy networks is the capability to dynamically determine the fuzzy rules from the given dataset. However, most of the existing fuzzy neural systems confronted some problems such as a priori computation to determine the number of clusters, inconsistent rule-base and heuristically defined node operations. Taking all deficiencies into consideration, a novel fuzzy rule transfer mechanism for self-constructing neural fuzzy inference networks, where transfer fuzzy rule is used as a substitute for the rule generation strategy of the SONFIN is proposed in this study. The proposed method not only promotes our learning process but also provides a stable and excellent performance. In order to demonstrate the feasibility and effectiveness of the proposed method, several examples, including the Mackey Glass time series prediction problem and a nonlinear dynamic system, are used to determine the network's performance. Experimental results demonstrate that the proposed method outperforms other methods on given sets of benchmark data with comparatively fewer rules.

The rest of the paper is organized as follows: Section 2 gives a brief introduction of SONFIN, FCM, CFC, PCFC and an overview on the proposed method and its architecture. Section 3 shows the experimental results on two different time-series datasets and finally conclusions are drawn in Section 4.

2. Proposed method

2.1 Self Constructing Neural Fuzzy Inference Networks

A self-constructing neural fuzzy inference networks (SONFIN) [18] was proposed by Juang and Lin, which has been applied to various applications [19-24]. The SONFIN always brings an effective network structure and speeds up the learning process with well-defined modeling capability compared to common neural networks.

The SONFIN consists of multiple layers, each of which has a finite fan-in of connections that are represented by weight values from other nodes and a fan-out of connections to other nodes. The function provides the net input for node is denoted as follows:

$$net-input = f[u_1^{(k)}, u_2^{(k)}, ..., u_n^{(k)}; w_1^{(k)}, w_2^{(k)}, ..., w_n^{(k)}] \tag{1}$$

where $u_1^{(k)}$, $u_2^{(k)}$, ….., $u_n^{(k)}$ are inputs to this node and $w_1^{(k)}$, $w_2^{(k)}$, ….., $w_n^{(k)}$ are the associated link weights. The superscript (k) indicates the layer number. The output of each node is an activation function value of its net input given by:

$$output = o_i^{(k)} = a(net-input) = a(f) \tag{2}$$

where a(.) denotes tha activation function. The functions of the nodes in each of the five layers of the SONFIN structure are briefly described as follows:

Layer 1: No computation is performed in this layer, the input values are directly transmited to the next layer.

$$f = u_i^{(1)} \text{ and } a^{(1)} = f \tag{3}$$

Layer 2: Using the Gaussian membership function the output of Layer 1 is calculated as follows:

$$f[u_{ij}^{(2)}] = -\frac{[u_{ij}^{(2)} - \mu_{ij}]^2}{\sigma_{ij}^2} \text{ and } a^{(2)} = e^f \tag{4}$$

where μ_{ij} is the mean and σ_{ij} is the variance of the Gaussian membership function of the ith input variable u_{ij} for the jth partition.

Layer 3: One fuzzy logic rule is represented by a node in this layer and it performs a precondition matching of a rule with an AND operation as follows:

$$f[u_i^{(3)}] = \prod_{i=1}^{n} u_i^{(3)} \text{ and } a^{(3)} = f \tag{5}$$

where n is the number of layer 2 nodes participating in the IF part of the rule.

Layer 4: Normalized firing strength is calculated in layer 3 and number of nodes in this layer is equal to that in Layer 3.

$$f[u_i^{(4)}] = \sum_{i=1}^{r} u_i^{(4)} \text{ and } a^{(4)}(f) = \frac{u_i^{(4)}}{f} \tag{6}$$

where r is the number of rule nodes in Layer 3.

Layer 5: The node integrates all the actions recommended in Layer 5 and acts as a defuzzifier. Each node in this layer corresponds to one output variable.

$$f[u_i^{(5)}] = \sum_i w_i u_i^{(5)},\ a^{(5)}(f) = f \tag{7}$$

For details on structure and parameter learning of the SONFIN, users can refer to [18].

2.2 Fuzzy C-means Clustering

In 1981, Bezdek introduced fuzzy c-means (FCM) [25], which allows each data point exhibits to one or more clusters that are specified by a membership function. The minimization of objective which decides the performance of FCM is defined as shown in Eq. (8).

$$J_M = \sum_{i=1}^{N} \sum_{j=1}^{c} u_{ij}^m \| x_i - v_j \|^2 \tag{8}$$

where M is real number great than 1, u_{ij} is the degree of membership of x_i in the cluster j, x_i is the i^{th} data point of d-dimension dataset, v_j is the d-dimension of the cluster and ||*|| is any norm expressing the similarity between any measured data and the center.

Procedure for FCM

1. Set up a value of c (number of clusters);
2. Select initial cluster prototype $V_1, V_2, \ldots, V_c$ from X_i, i=1, 2, …, N;
3. Compute the distance $\|X_i - V_j\|$ between objects and prototypes;
4. Compute the elements of the fuzzy partition matrix (i=1, 2, …, N; j=1, 2, …, c)

$$u_{ij} = \left[\sum_{l=1}^{c} \left(\frac{\|x_i - v_j\|}{\|x_i - v_l\|} \right) \right]^{-1} \tag{9}$$

5. Compute the cluster prototypes (j=1, 2, …, c)

$$V_j = \frac{\sum_{i=1}^{N} u_{ij}^2 x_i}{\sum_{i=1}^{N} u_{ij}^2} \tag{10}$$

6. Stop if the convergence is attained or the number of iterations exceeds a given limit. Otherwise, go to step 3

2.3 Collaborative Fuzzy Clustering

Pedrycz introduced the collaborative fuzzy clustering (CFC) technique [49, 50], in which several subsets of patterns can be processed together with an objective to finding a structure that is common to all of them. Horizontal collaborative clustering and vertical collaborative clustering are the two major variants of the CFC and it has been applied in different research areas to solve clustering and modeling problems [26, 51-53]. The general schemes of horizontal collaborative clustering and vertical collaborative clustering are shown in Fig. 1 and Fig. 2 respectively.

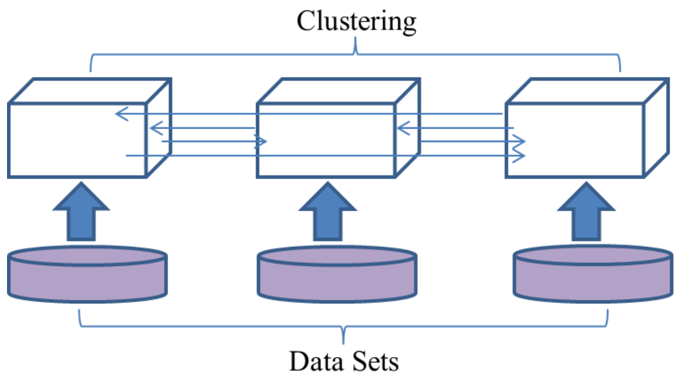
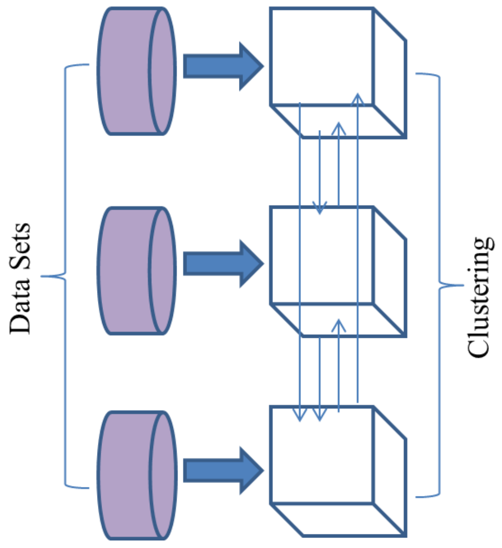


Fig. 1. A General scheme of horizontal clustering



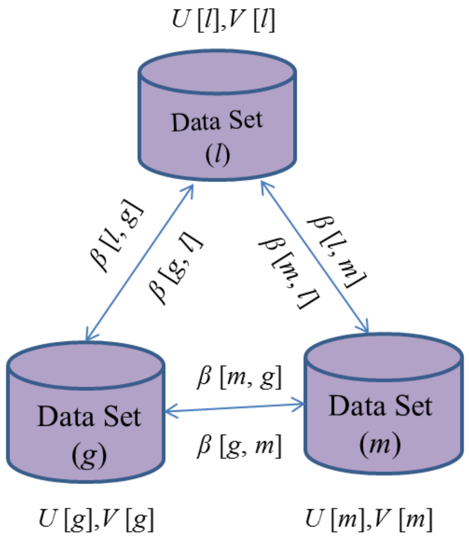


Fig. 2. A General scheme of vertical clustering

Fig. 3. Collaborative clustering scheme

The objective function for collaboration technique is given by:

$$Q[l] = \sum_{i=1}^{N}\sum_{j=1}^{c} u_{ij}^2[l] d_{ij}^2[l] + \sum_{\substack{m=1 \\ m \neq l}}^{p} \beta[l,m] \sum_{i=1}^{N}\sum_{j=1}^{n} \{u_{ij}[l] - u_{ij}[m]\}^2 d_{ij}^2[l] \tag{11}$$

where β is a user defined parameter based on datasets (β>0), $\beta[l, m]$ denotes the collaborative coefficient with collaborative effect on dataset l through m, c is the number of clusters. l=1, 2, ….., p. p is the number of datasets, N is the number of patterns in the dataset, u represents the partition matrix, n is the number of features, and d is an Euclidean distance between patterns and prototypes.

Fig. 3 shows the connections of matrices in order to accomplish the collaboration between the subsets of the dataset. First, we solve the problem for each dataset separately and allow the results to interact globally by forming a collaborative process between the datasets. Collaborative fuzzy partitioning is carried out through an iterative optimization of the objective function as shown in Eq. (11) with an update of partition matrix $u[l]$ and the prototype $v_i[l]$. For optimization details please refer to [49, 50]. The prototype and partition matrices bring the way of structural findings at each dataset in the structure learning phase of the proposed method. The calculations of the partition matrix $u[l]$ and the prototype $v_i[l]$ are as follows:

$$u_{st}[l] = \frac{\varphi_{st}[l]}{1+\psi[l]} + \frac{1}{\sum_{j=1}^{c} \frac{d_{st}^2[l]}{d_{jt}^2[l]}} \left[1 - \sum_{j=1}^{c} \frac{\varphi_{jt}[l]}{1+\psi[l]} \right] \tag{12}$$

where

$$\varphi_{st}[l] = \sum_{\substack{m=1 \\ m \neq l}}^{p} \beta[l,m] u_{st}[m] \tag{13}$$

$$\psi[l] = \sum_{\substack{m=1 \\ m \neq l}}^{p} \beta[l,m] \tag{14}$$

and

$$v_{st}[l] = \frac{\sum_{k=1}^{N} u_{sk}^2[l] x_{kt}[l] + \sum_{\substack{m=1 \\ m \neq l}}^{p} \beta[l,m] \sum_{k=1}^{N} (u_{sk}[l] - u_{sk}[m])^2 x_{kt}[l]}{\sum_{k=1}^{N} u_{sk}^2[l] + \sum_{\substack{m=1 \\ m \neq l}}^{p} \beta[l,m] \sum_{k=1}^{N} (u_{sk}[l] - u_{sk}[m])^2} \tag{15}$$

Procedure for CFC

```
1. Given, subsets of patterns $X_1, X_2, \ldots, X_p$.
2. Select, distance function, number of clusters ($c$), termination condition and collaboration
   coefficient $\beta[l, m]$.
3. Compute, randomly initialize all partition matrices $U[1], U[2], \ldots, U[P]$
4. Phase I
       For each data
           Repeat
               Compute, prototype { $V_j[l]$, $j=1, 2, \ldots, c$ and partition matrices $U[l]$ for all
               subsets of patterns}
           Until a termination condition has been satisfied
   End of Phase I
5. Phase II
           Repeat
               For the matrix of collaborative links $\beta[l, m]$.
               Compute, prototype $V_j[l]$ and partition matrices $U[l]$ by using (12) and (15).
           Until a termination condition has been satisfied
    End of phase II
```

2.4 Preprocessed Collaborative Fuzzy Clustering

A preprocessed collaborative fuzzy clustering (PCFC) [54] is an improved version of the collaborative fuzzy clustering (CFC). In this section, the problem, which lies with CFC has pointed out and an appropriate solution has been given by proposing a cluster center mapping mechanism before it goes for collaboration process.

2.4.1 Problem with CFC

By taking a direct subtraction of two different partition matrices $u_{ik}[l]$ and $u_{ik}[m]$ (i.e. $u_{ik}[l]$-$u_{ik}[m]$), we may lose the useful information under different partition matrices of one pattern X_k of the same cluster. The cluster described by k^{th} row $V_k[l]$ in $u_{ik}[l]$ may be different from the one described by the k^{th} row in $V_k[m]$ in $u_{ik}[m]$. If the rows order of one matrix changes, the subtraction between two matrices changes as well. In this case, taking direct subtraction between two matrices $u_{ik}[l]$ and $u_{ik}[m]$ is not an appropriate way.

2.4.2 Solution by PCFC

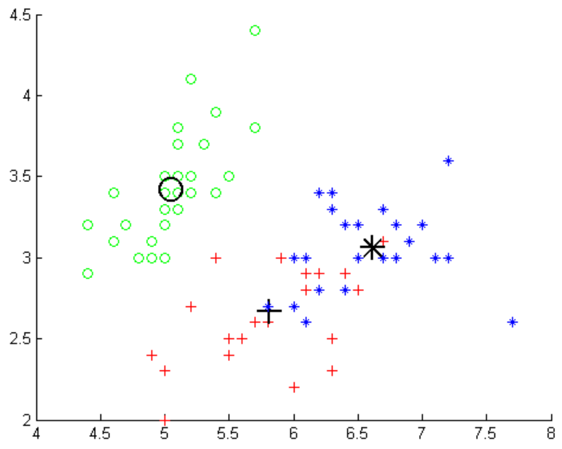
To find a constructive approach of preprocessing in order to rearrange the row order of $u_{ik}[l]$ corresponding to the row order of $u_{ik}[m]$ in a rational way, the matching row pair is determined as follows:

$$r = \arg \min_{j=1,2..,c} \sum_{i=1}^{n} (V_{ki}[l] - V_{ji}[m])^2 \tag{16}$$

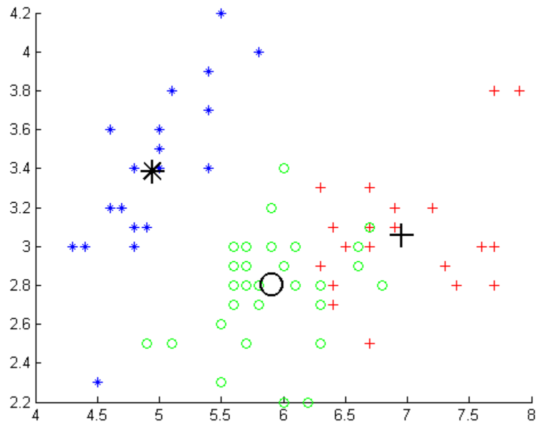
The k^{th} row of $V[l]$ and the r^{th} row of $V[m]$ are considered to be a matching row pair (k = 1, 2, …, c), where the number of features is denoted by n. Similarly, update $u_{ik}[l]$ and $u_{ik}[m]$ with correspond to $V[l]$ and $V[m]$.

2.4.3 Discussions

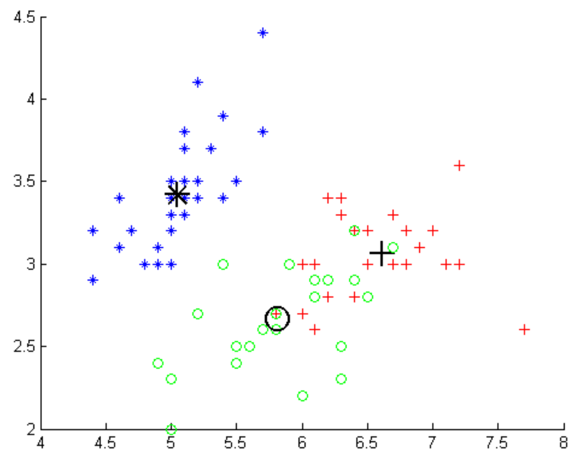
In order to verify the mapping mechanism, we have used the paradigm of three classes and then divided equally into two subsets of dataset, namely *dataset1* and *dataset2*. Fig. 4 (a) and 4 (b) are clustered feature vectors of *dataset1* and *dataset2*, respectively. As we can see, in fig. 4(a) and 4(b), the first cluster (green color) of *dataset1* matches with the second cluster of *dataset2*, the second cluster (red color) of *dataset1* matches with the third cluster of *dataset2* and the third cluster (blue color) of *dataset1* matches with the first cluster of *dataset2*, which are totally mismatched with each other. Fig. 4(c) and 4 (d) show the plotting results after the mapping mechanism, here we can see the effect of centroid mapping for prototype and row order mapping with the partition matrix. Now, we can easily take the difference(s) between rows of *dataset1* and *dataset2* and easily do mapping between them.



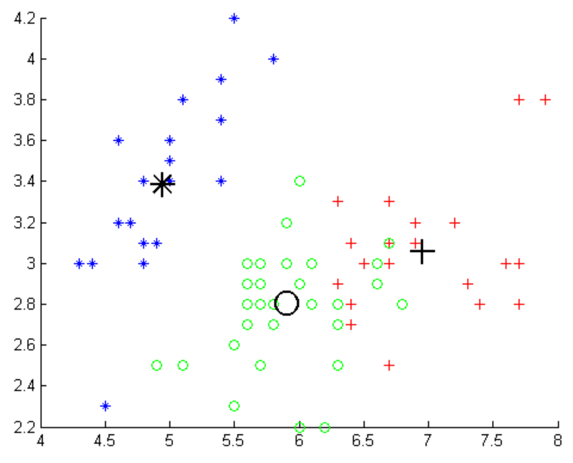
(a) Clustered feature vectors of dataset1 based on FCM



(b) Clustered feature vectors of dataset2 based on FCM



(c) Clustered feature vectors of dataset1 based on FCM after mapping



(d) Clustered feature vectors of dataset2 of FCM after mapping

Fig. 4 Clustered feature vectors of dataset1 and dataset2

2.4.4 Algorithm for PCFC

Based on the above discussions and the results, we have added one more phase called phase III in the CFC procedure for preprocessing and the refined algorithm is as follows:

Procedure for PCFC

```
1. Given, subsets of patterns $X_1, X_2, \ldots, X_p$.
2. Select, distance function, number of clusters ($c$), termination condition, and collaboration
   coefficient $\beta[l, m]$.
3. Compute, randomly initialize all partition matrices $U[1], U[2], \ldots, U[P]$
4. Phase I
       For each data
           Repeat
               Compute, prototype { $V_j[l]$, $j$=1, 2, …, $c$ and partition matrices $U[l]$ for all
               subsets of patterns}
           Until a termination condition has been satisfied
       Communicate cluster prototype from each data site to all others;
   End of Phase I
5. Phase II
       Choose an approach as given in Eq. (16) for the preprocessing on cluster prototype and
       its corresponding partition matrices in order to adjust the feature row order.
   End of Phase II
6. Phase III
       Repeat
           For the matrix of collaborative links $\beta[l, m]$.
           Compute, prototype $V_j[l]$ and partition matrices $U[l]$ by using (12) and (15).
       Until a termination condition has been satisfied
   End of Phase III
```

2.5 Rule-transfer-based Self Constructing Neural Fuzzy Inference Networks

The proposed method combines the collective knowledge of the PCFC and neural fuzzy networks which possesses the advantages of both neural networks and fuzzy systems. The structure and parameter learning parts of fuzzy neural networks can be identified with the knowledge of the PCFC and learning ability of neural networks, which improves the interpretation result of fuzzy neural networks. The proposed method only uses one halve of the patterns of the given dataset to model the system whereas other methods use the entire datasets to achieve similar performance. Therefore, we can say that the proposed method is capable of solving the big data issues with satisfactory result in terms of computational complexity due to fewer rules while taking into consideration of privacy and security of the datasets. The framework structure of proposed method is shown in Fig. 5.

The proposed method is divided into two phases, namely the structure learning phase and the parameter learning phase. In structure learning phase, the collective knowledge of the PCFC is applied for building the network structure in order to replace the self-evolving ability of the SONFIN. The parameter leaning phase helps the network to redefine their parameter values based on the knowledge coming from structure learning phase. The performance of the proposed method is demonstrated on the Mackey glass time series prediction problem with two different conditions and a nonlinear dynamics system identification problem. The proposed method is analyzed on the given datasets under MATLAB 7.9 and implemented using an Intel *i*5, 3.1 GHz CPU, with 4 GB of RAM running on Windows XP 7 (32 bit) operating system.

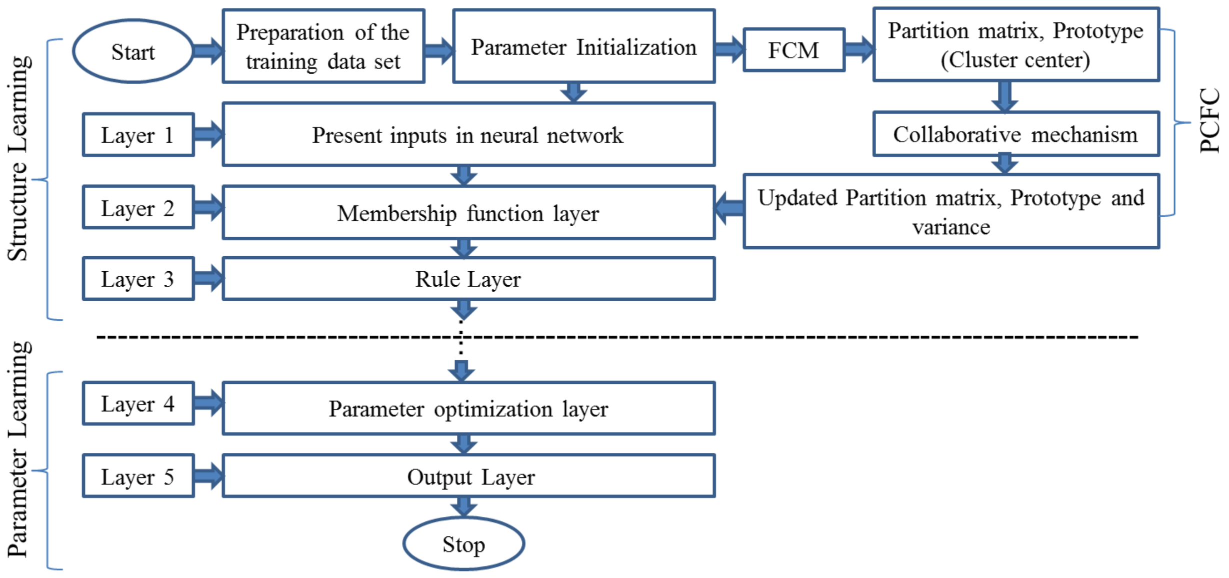

Structure Learning
Start
Preparation of the training data set
Parameter Initialization
FCM
Partition matrix, Prototype (Cluster center)
Layer 1
Present inputs in neural network
Collaborative mechanism
PCFC
Layer 2
Membership function layer
Updated Partition matrix, Prototype and variance
Layer 3
Rule Layer
Parameter Learning
Layer 4
Parameter optimization layer
Layer 5
Output Layer
Stop

Fig. 5. Framework of the proposed method

The results of the proposed method are compared with other previously proposed methods, such as GEBF-OSFNN [14], RBF-AFS [27], OLS [28], DFNN [29], FAOS-PFNN [30], RAN [31], RANEKF [32], MRAN [33], GGAP-RBF [34], OS-ELM [35], SOFNN [36], SOFNNGA [37], Khayat's model [38], SAFIS [39], eTS [40], Mean shift method [41], KNN method [41], Space partitioning method [41], and simpleLeTS [42]. Based on the experimental results shown in Tables 1-4, it can be easily seen that the proposed method outperforms other models on the benchmark problems.

The performance of the proposed method is evaluated in terms of root mean square error (RMSE) and the number of rule during training and testing phase. The RMSE is a popular and useful index for assessing the performance of the predictors [43] and is given by Eq. (17).

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{17}$$

where n is the number of predictions, y is the desired value and $\hat{y}$ is the predicted value.

3. Experimental Results

3.1 Mackey Glass Time Series Prediction

One of the classical benchmark chaotic Mackey glass time-series predictions used in [44-48] is chosen for verification of the proposed method. The discrete model of the time series is as follows:

$$x(t+1) = (1-a)x(t) + \frac{bx(t-\tau)}{1+x^{10}(t-\tau)} \tag{18}$$

where a=0.1, b=0.2, τ=17 and x(0)=1.2. The problem is to predict the value $x(t+p)$ (where p=6) from the following prediction model:

$$x(t+p) = f\{x(t), x(t-6), x(t-12), x(t-18)\} \tag{19}$$

A set of 1000 samples extracted from Eq. (18) for each, training and testing purpose and these samples used for preparing the input and output sample data based on Eq. (19). Subsequently, training dataset is divided into two datasets, namely *dataset1* and *dataset2*, which contain 500 patterns each. The proposed method only uses the collective knowledge of 500 patterns of *dataset1*/*dataset2* for neural networks training after applying the PCFC mechanism. The parameters used for the proposed method in this prediction model are: P1=0.7 (learning rate), P2=0.5 (collaboration coefficient) and P3=500 (number of iterations). Table 1 shows a performance comparison of the proposed method with GEBF-OSFNN, RBF-AFS, OLS, DFNN and FAOS-PFNN. The performance of the proposed method as shown in Table 1 is the mean value based on 10 experimental trials. The best training and testing RMSE value during 10 experimental trials is 0.0005 and 0.0012, respectively. It can be easily seen that the proposed method achieves better performance while using significantly fewer rules. Fig. 6 and 8 show the output value of predicted and desired model and Fig. 7 and 9 show the predicted errors during training and testing phase, respectively. It can be easily seen from Table 1 that the performance achieved by the proposed method is superior to the previously proposed methods.

Table 1
Performance comparison of DDNFS-CFCM, GEBF-OSFNN, RBF-AFS, OLS, DFNN and FAOS-PFNN

Method	No. of Rules	Training RMSE	Testing RMSE
RBF-AFS [27]	21	0.0107	0.0128
OLS [28]	13	0.0158	0.0162
FAOS-PFNN [30]	11	0.0073	0.0127
GEBF-OSFNN [14]	10	0.0091	0.0087
DFNN [29]	10	0.0082	0.0127
DDNFS-CFCM	**6**	**0.0009**	**0.0034**

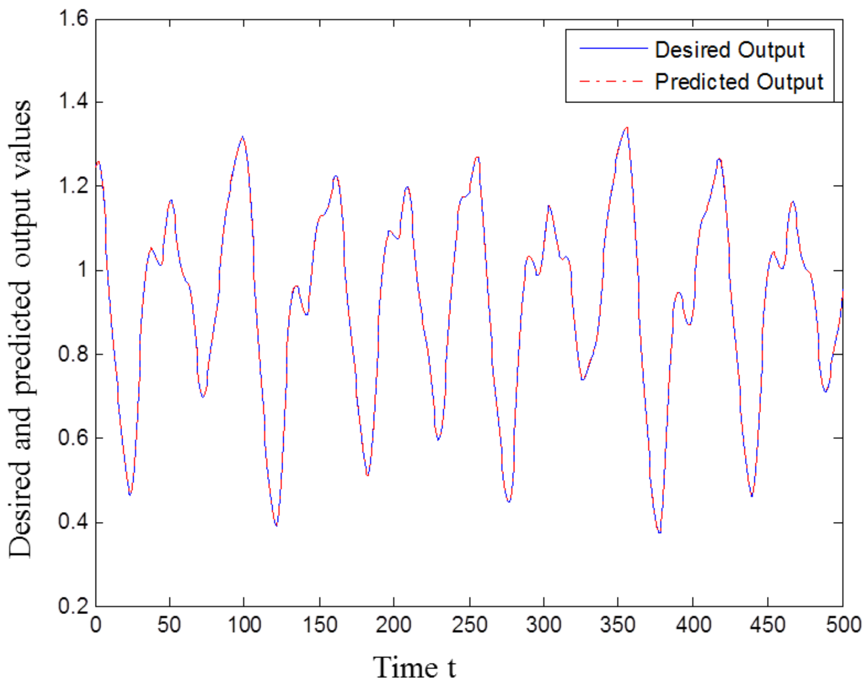


Fig. 6. Desired and predicted outputs during training

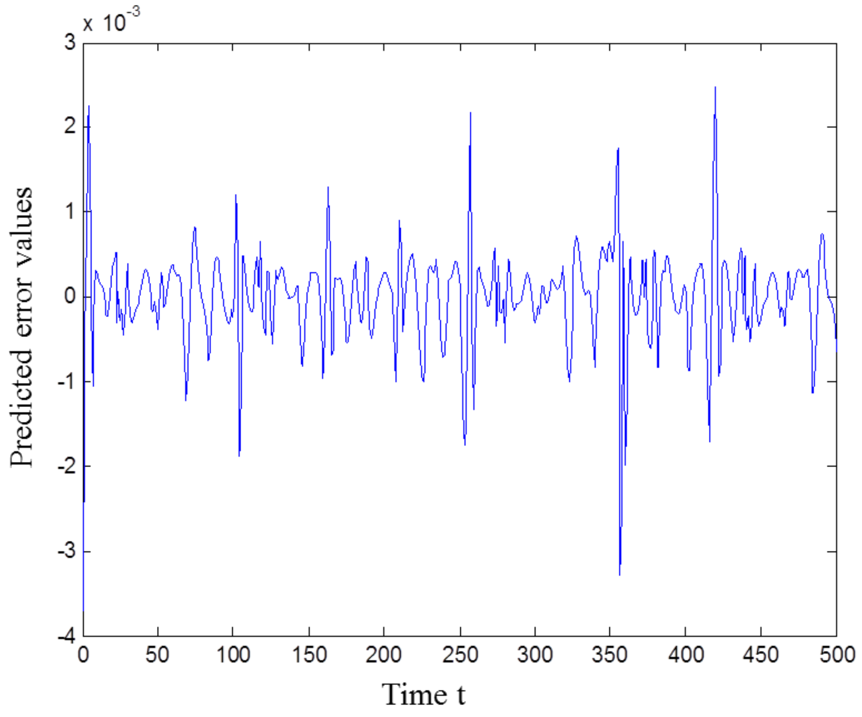


Fig. 7. Predicted error during training

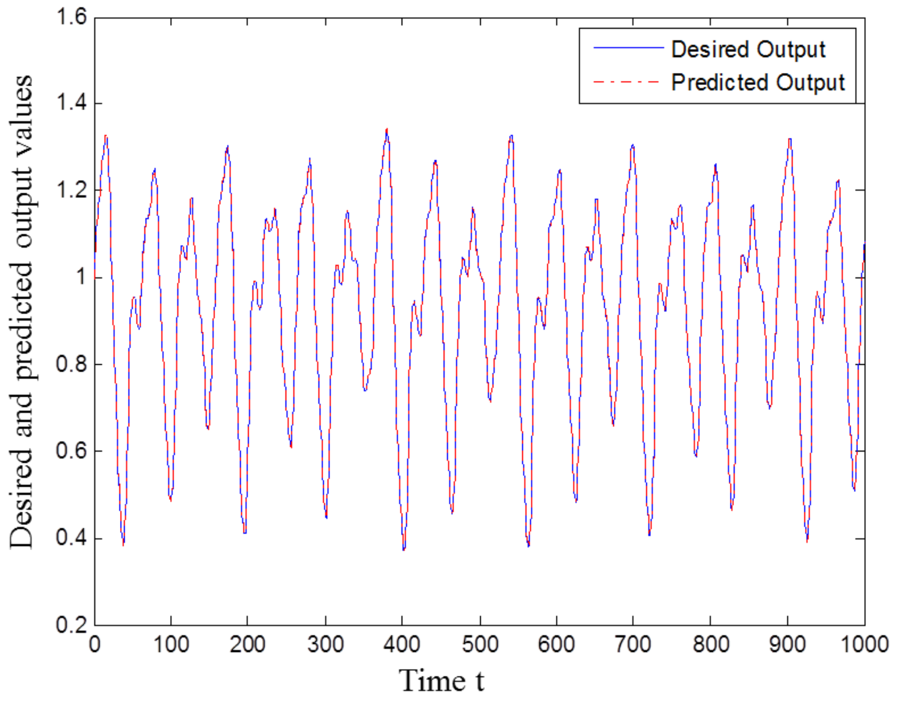


Fig. 8. Desired and predicted outputs during testing

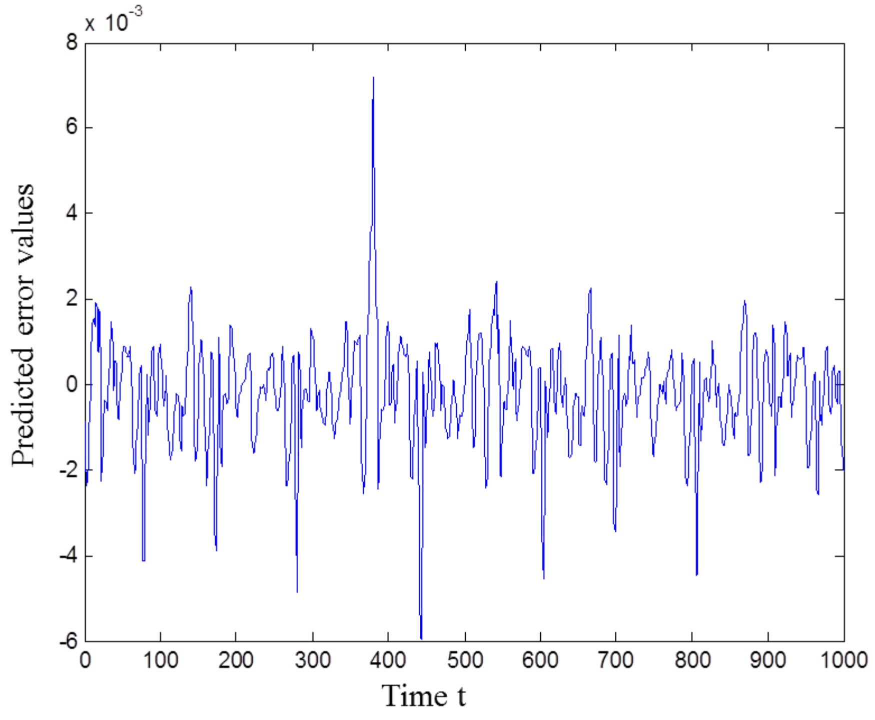


Fig. 9. Predicted error during testing

Table 2 shows the performance of the proposed method with initial condition *x*(0)=0.3 and *p*=50. The parameters used for the proposed method in this prediction model are: *P*1=0.04 (learning rate), *P*2=0.5 (collaboration coefficient) and *P*3=500 (number of iterations). Based on these values, the proposed method also achieves better results in terms of RMSE. A comparison of the proposed method with GEBF-OSFNN, RAN, RANEKF, MRAN, GGAP-RBF, OS-ELM and FAOS-PFNN is given in Table 2. The performance of the proposed method as shown in Table 2 is the mean value based on 10 experimental trials. The best training and testing RMSE value during 10 experimental trials is 0.0083 and 0.0172, respectively. Fig. 10 and 12 show the desired and predicted outputs and Fig. 11 and 13 show the predicted errors during the training and testing phase, respectively. It can be seen that the proposed method outperforms the previous proposed methods while requiring significant fewer rules.

Table 2
Performance comparison of DDNFS-CFCM, GEBF-OSFNN, RAN, RANEKF, MRAN, GGAP-RBF, OS-ELM and FAOS-PFNN

Method	No. of Rules	Training RMSE	Testing RMSE
OS-ELM [35]	120	0.0184	0.0186
RAN [31]	39	0.1006	0.0466
RANEKF [32]	23	0.0726	0.0240
MRAN [33]	16	0.1101	0.0337
GGAP-RBF [34]	13	0.0700	0.0368
DDNFS-CFCM	**9**	**0.0105**	**0.0260**

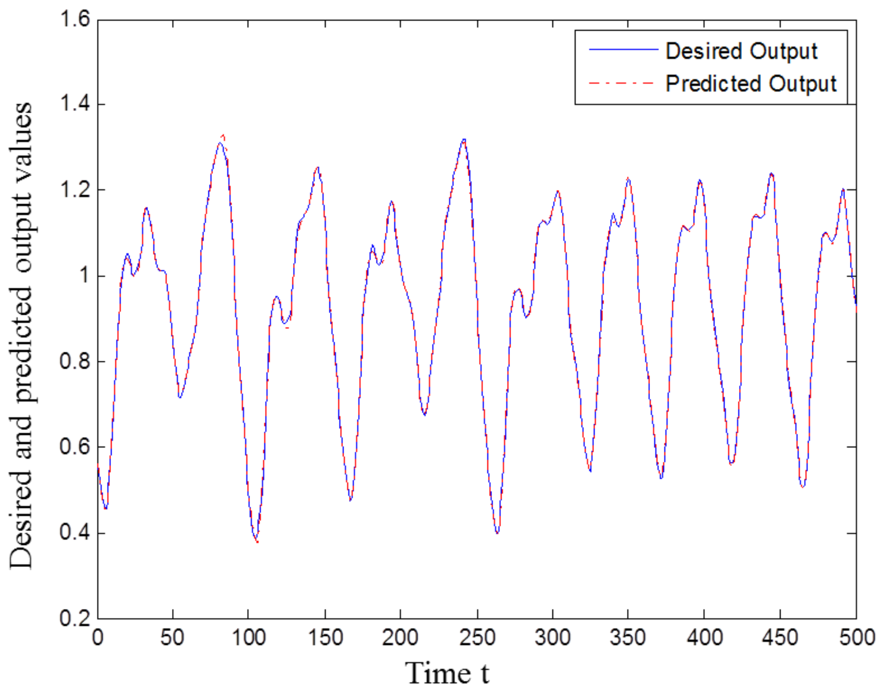


Fig. 10. Desired and predicted outputs during training

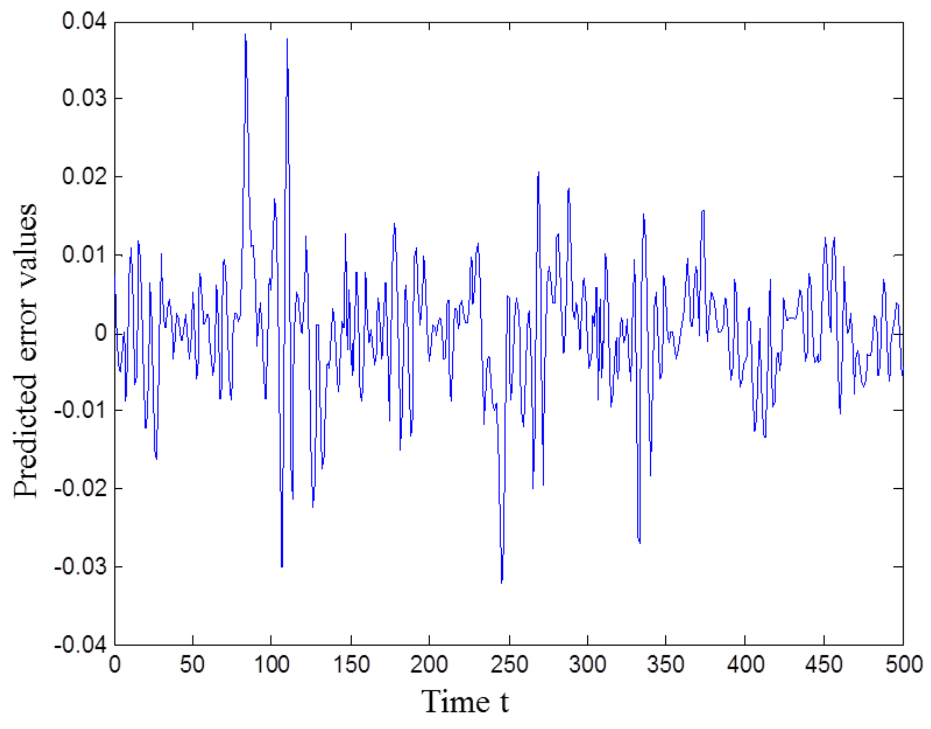


Fig. 11. Predicted error during training

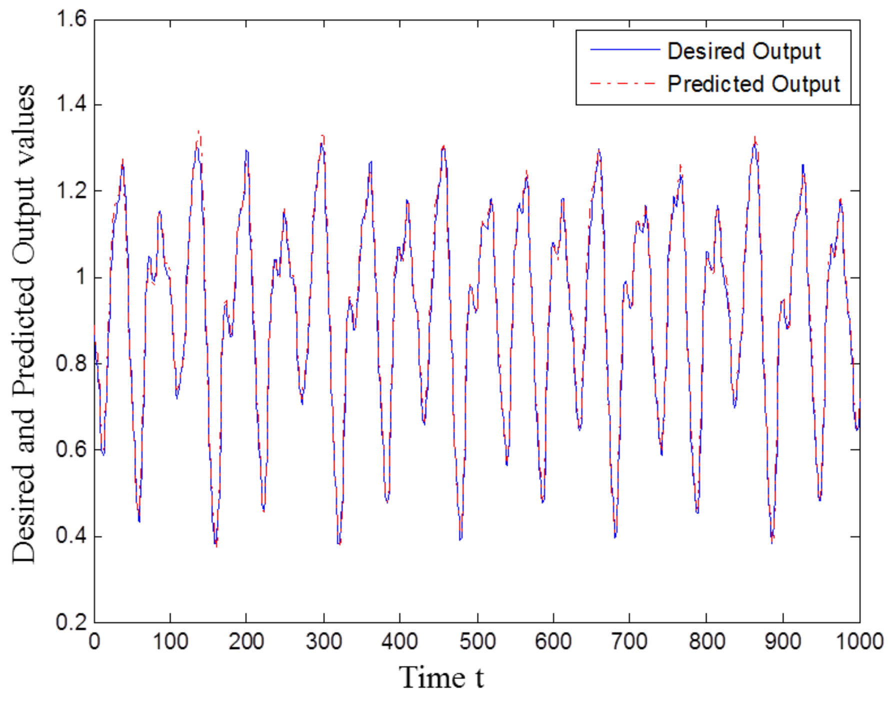


Fig. 12. Desired and predicted outputs during testing

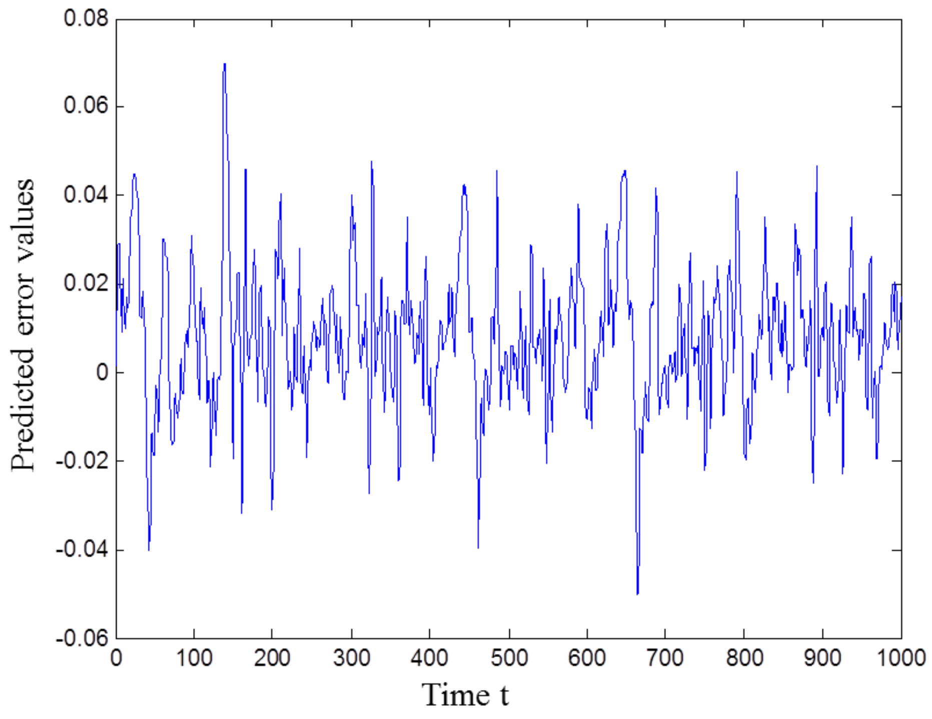


Fig. 13. Predicted error during testing

3.2 *Nonlinear dynamics system identification problem* I

The plant to be identified is described in Eq. (20):

$$y(t+1) = \frac{y(t)y(t-1)[y(t)+2.5]}{1+y^2(t)+y^2(t-1)} + u(t) \tag{20}$$

If a series-parallel identification model is used for identifying the plant, the model can be described by Eq. (21)

$$\hat{y}(t+1) = f\{y(t), y(t-1), u(t)\} \tag{21}$$

where the input $u(t)=\sin(2\pi t/25)$, $y(t+1)$ is the output and this network contains three inputs and one output. The initial input values $y(0)=0$ and $y(1)=0$ was used. A set of 200 data is generated for each, training and testing dataset. Subsequently, training dataset is divided into two datasets, namely *dataset1* and *dataset2*, which contain 100 patterns each. The proposed method only uses the collective knowledge of 100 patterns of *dataset1*/*dataset2* for neural networks training after applying the PCFC mechanism. The parameters used for the proposed method in this prediction model are: $P1$=0.2 (learning rate), $P2$=0.5 (collaboration coefficient) and $P3$=500 (number of iterations). The performance of the proposed method as shown in Table 3 is the mean value based on 30 experimental trials. The best training and testing RMSE value during 30 experimental trials is 0.0023 and 0.0020, respectively. Table 3 shows the performance comparison of the proposed method with Mean-Shift method, KNN method, Space partitioning method, Khayat's model, SOFNNGA, and SOFNN. Fig. 14 and 16 show the predicted and desired output values and Fig. 15 and 17 show the predicted errors during the training and testing phase, respectively.

Table 3
Performance comparison of DDNFS-CFCM, OSFNN, SOFNNGA, Khayat's model, Mean shift method, KNN method and Space partitioning method

Method	No. of Rules	Training RMSE	Testing RMSE
Space partitioning method [41]	9	0.0065	0.0055
OSFNN [36]	5	0.0157	0.0151
Mean shift method [41]	5	0.0137	0.0127
SOFNNGA [37]	4	0.0159	0.0146
Khayat's model [38]	4	0.0147	0.0141
KNN method [41]	4	0.0150	0.0131
DDNFS-CFCM	4	0.0036	0.0031

3.3 Nonlinear dynamics system identification problem II

The nonlinear system is expressed as follows:

$$y(t+1) = \frac{y(t)}{1+y^2(t)} + u^3(t) \tag{20}$$

where $u(t)$ is the input signal, which is generated by using the sinusoidal function given by $u(t)$=sin(2πt)/100. A set of 200 data is generated for each, training and testing dataset. Subsequently, training dataset is divided into two datasets, namely *dataset1* and *dataset2*, which contain 100 patterns each. The proposed method only uses the collective knowledge of 100 patterns of *dataset1*/*dataset2* for neural networks training after applying the PCFC mechanism.

The parameters used for the proposed method in this prediction model are: $P1$=0.2 (learning rate), $P2$=0.5 (collaboration coefficient) and $P3$=300 (number of iterations). The proposed method only uses 30000 data points (100 data patterns with 300 numbers of iteration) to train the system whereas, the other methods shown in Table 4 use 50000 data points. The inputs $y(t)$ and $u(t)$ follow the uniform sample distribution in the interval [-1.5, 1.5] and [-1.0, 1.0] respectively. The performance of the proposed method as shown in Table 4 is the mean value based on 30 experimental trials. The best testing RMSE value during 30 experimental trials is 0.0045. Table 4 shows a performance comparison of the proposed method with SAFIS, eTS, OS-fuzzy-ELM, and simpleLeTS. Fig. 18 and 19 show the predicted and desired output values and predicted errors during the testing phase, respectively. It can be easily seen that the proposed method outperforms other methods in terms of RSMS while keeping significantly fewer rules.

Table 4
Performance comparison of DDNFS-CFCM, SAFIS, MRAN, RANEKF, simpleLeTS, and eTS

Method	No. of Rules	Testing RMSE
eTS [40]	19	0.0082
simpleLeTS [42]	18	0.0122
RANEKF [32]	11	0.0184
MRAN [33]	10	0.0129
SAFIS [39]	8	0.0116
DDNFS-CFCM	5	0.0046

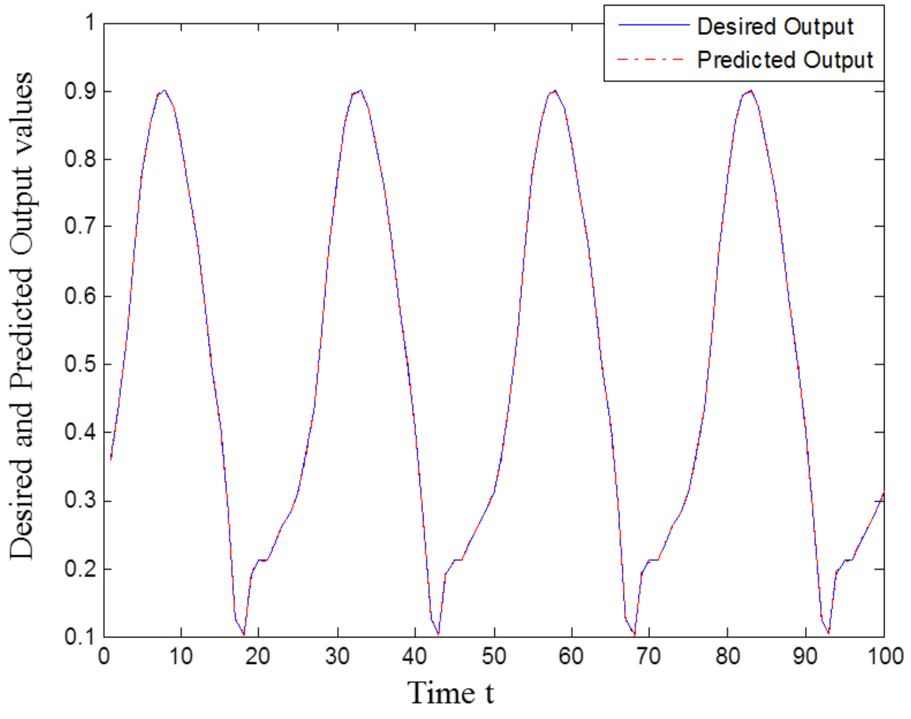


Fig. 14. Desired and predicted outputs during training

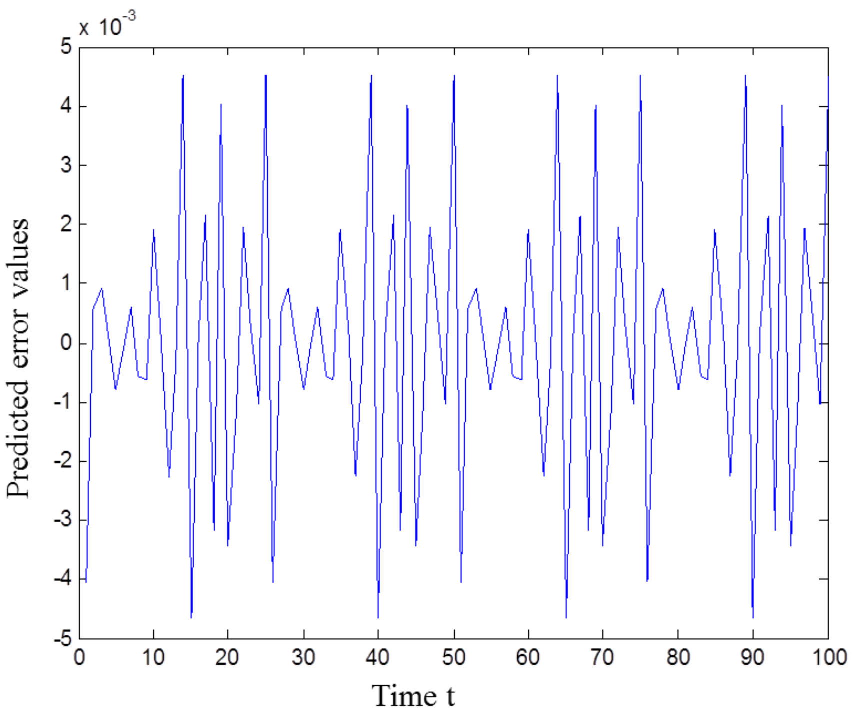


Fig. 15. Predicted error during training

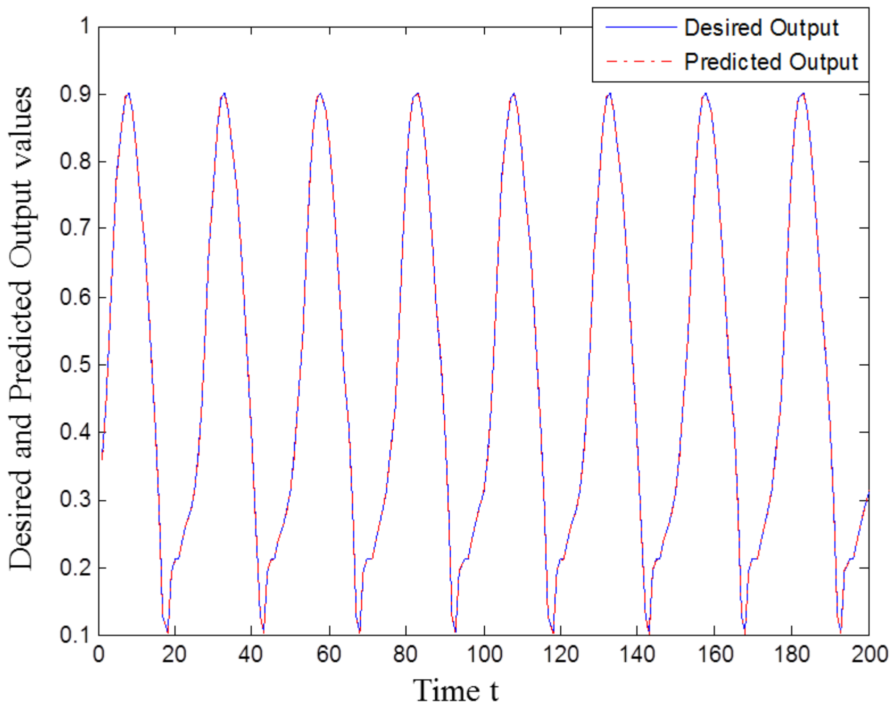


Fig. 16. Desired and predicted outputs during testing

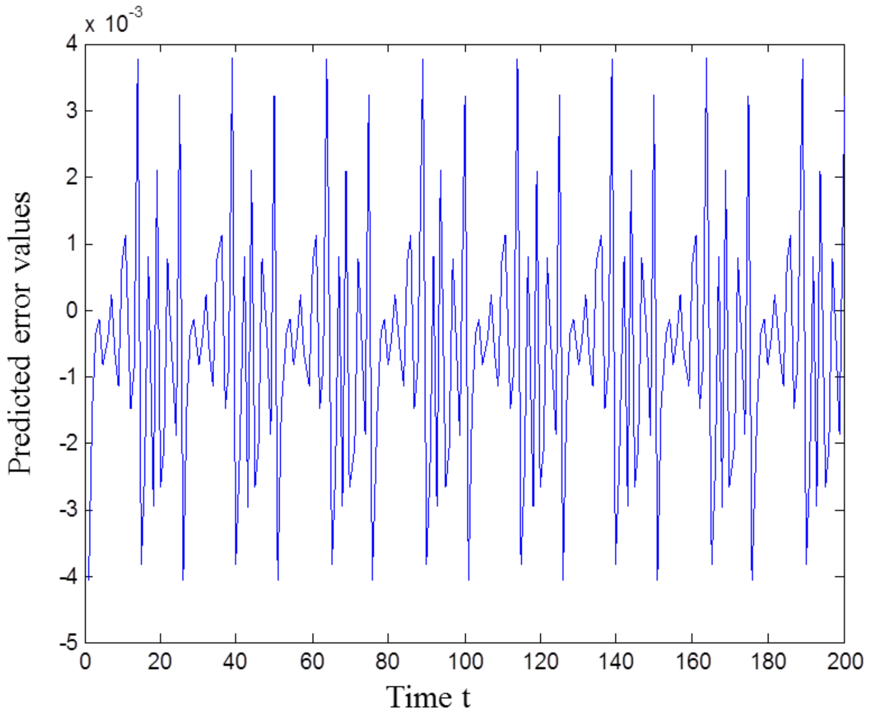


Fig. 17. Predicted error during testing

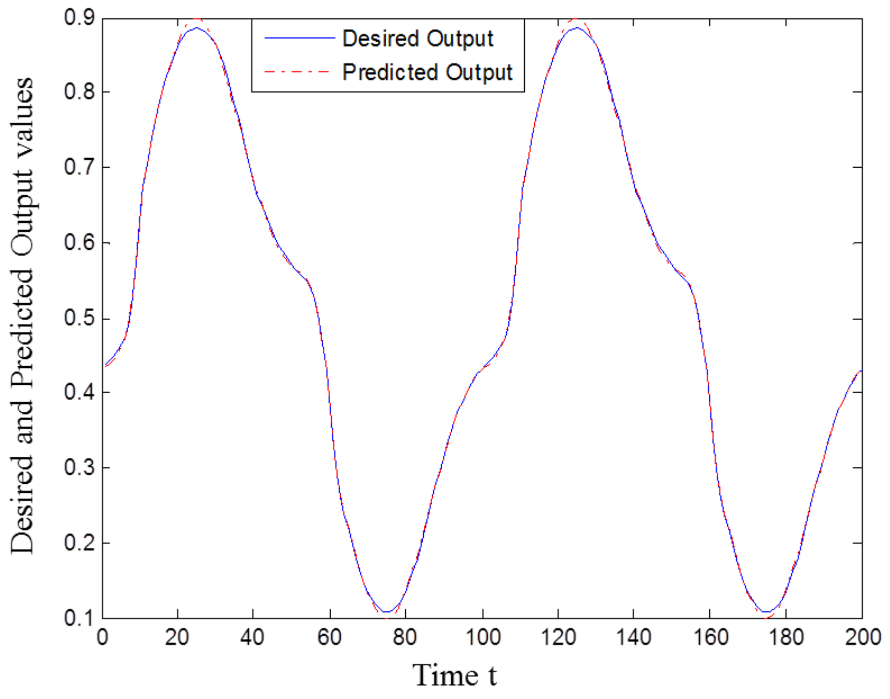


Fig. 18. Desired and predicted outputs during testing

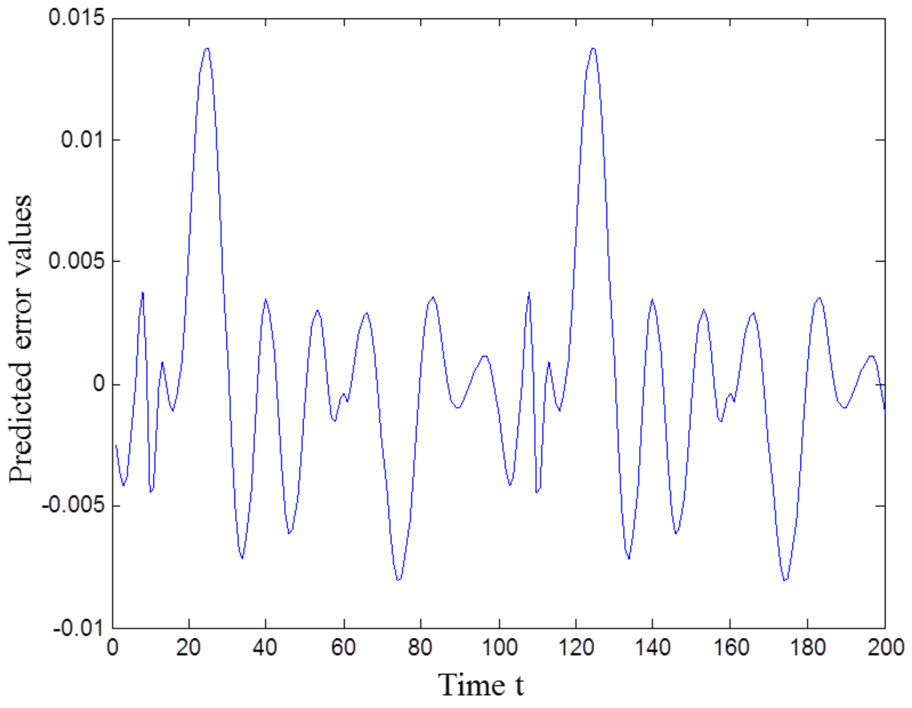


Fig. 19. Predicted error during testing

4. CONCLUSIONS

In this paper, a novel fuzzy rule transfer mechanism for self-constructing neural fuzzy inference networks is proposed. The features of the proposed method are: (1) Fuzzy rules are generated facilely by fuzzy c-means (FCM) and then adapted by the preprocessed collaborative fuzzy clustering (PCFC) technique, and (2) Structure and parameter learning are performed simultaneously without selecting initial parameters. Based on the experimental results, the proposed method has shown satisfactory results in terms of computational complexity with significantly fewer rules while taking into consideration of privacy and security of the datasets. The proposed method is superior to existing state-of-the-art methods as demonstrated on the set of benchmark problems.

ACKNOWLEDGEMENT

The authors would like to thank the Editor-in-Chief, Associate Editor and anonymous referees for their valuable comments and suggestions in enhancing the quality of this paper. This work was supported in part by the UST-UCSD International Center of Excellence in Advanced Bio-engineering sponsored by the Taiwan National Science Council I-RiCE Program under Grant Number: MOST 103-2911-I-009-101, in part by MOST 103-2627-E-009-001, in part by the Aiming for the Top University Plan, under Contract: 104W963, and in part by the Army Research Laboratory: W911NF-10-2-0022.

References

[1] C. T. Lin and C. S. G. Lee, Neural Fuzzy Systems: a Neural-Fuzzy Synergism to Intelligent Systems, Englewood Cliffs, NJ: Prentice-Hall, (1996).
[2] L. X. Wang, Adaptive Fuzzy Systems and Control: Design and Stability Analysis, Prentice-Hall, Englewood Cliffs (1994).
[3] J. S. R. Jang, C. T. Sun, and E. Mizutani, Neuro-fuzzy and Soft Computing, Prentice-Hall, Englewood Cliffs, NJ, (1997).
[4] K. B. Cho and B. H. Wang, Radial basis function based adaptive fuzzy systems and their applications to system identification and prediction, Fuzzy Sets and Systems, vol. 83, no.3, pp. 325-339, (1996).
[5] K. Tanaka, M. Sano, and H. Watanabe, Modeling and control of carbon monoxide concentration using a neuro-fuzzy technique, IEEE Transactions on Fuzzy Systems, vol. 3, no. 3, pp. 271-279, (1995).
[6] J. J. Buckley and Y. Hayashi, Fuzzy neural networks: A survey, Fuzzy Sets and Systems, vol. 66, no. 1, pp. 1–13, (1994).
[7] J. J. Shann and H. C. Fu, A fuzzy neural networks for rule acquiring on fuzzy control systems, Fuzzy Sets and Systems, vol. 71, no. 3, pp. 345–357, (1995).
[8] A. Abraham, Neuro Fuzzy Systems: State-of-the-art Modeling Techniques, Connectionist Models of Neurons, Learning Processes, and Artificial Intelligence Lecture Notes in Computer Science, vol 2084, pp 269-276, (2001).
[9] H. J. Rong, N. Sundararajan, G. B. Huang, and P. Saratchandran, Sequential Adaptive Fuzzy Inference System (SAFIS) for nonlinear system identification and prediction, vol. 157, no. 9, pp. 1260-1275, (2006).
[10] D. Dovzan and I. Skrjanc, Recursive fuzzy c-means clustering for recursive fuzzy identification of time-varying processes, vol. 50, no. 2, pp. 159-169, (2011).
[11] J. S. Wang and C. S. G. Lee, Self-Adaptive Neuro-Fuzzy Inference Systems for Classification Applications, IEEE Transaction on Fuzzy Systems, vol. 10, no. 6, pp. 790-802, (2002).
[12] G. Leng, G. Prasad, and T. M. McGinnity, An on-line algorithm for creating self-organizing fuzzy neural networks, Neural Network, vol. 17, no. 10, pp. 1477-1493, (2004).
[13] M. J. Er and S. Wu, A fast learning algorithm for parsimonious fuzzy neural systems, Fuzzy Sets and Systems, vol. 126, no. 3, pp. 337-351, (2002).
[14] N. Wang, A Generalized Ellipsoidal Basis Function Based Online Self-constructing Fuzzy Neural Network, Neural Processing Letters, vol. 34, no. 1, pp. 13-37, (2011).

[15] H. Han and J. Qiao, A Self-Organizing Fuzzy Neural Network Based on a Growing-and-Pruning Algorithm, IEEE Transaction on Fuzzy Systems, vol. 18, no. 6, pp. 1129-1143, (2010.)
[16] C. J. Lin and C. H. Chen, Nonlinear system control using self-evolving neural fuzzy inference networks with reinforcement evolutionary learning, Applied Soft Computing, vol. 11, no. 8, pp. 5463-5476, (2011).
[17] H. Malek, M. M. Ebadzadeh, and M. Rahmati, Three new fuzzy neural networks learning algorithms based on clustering, training error and genetic algorithm, Applied Intelligence, vol. 37, no. 2, pp. 280-289, (2012).
[18] C. F. Juang and C. T. Lin, An On-Line Self-Constructing Neural Fuzzy Inference Network and Its Applications, IEEE Transaction on Fuzzy Systems, vol. 6, no. 1, pp. 12-32, (1998).
[19] C. T. Lin, S. F. Tsai, and L. W. Ko, EEG-Based Learning System for Online Motion Sickness Level Estimation in a Dynamic Vehicle Environment, IEEE Transaction on Neural Network and Learning Systems, vol. 24, no. 10, pp. 1689-1700, (2013).
[20] C. F. Juang, T. C. Chen, and W. Y. Cheng, Speedup of Implementing Fuzzy Neural Networks With High-Dimensional Inputs Through Parallel Processing on Graphic Processing Units, IEEE Transaction on Fuzzy Systems, vol. 19, no. 4, pp. 717-728, (2011).
[21] R. C. Wu, C. T. Lin, S. F. Liang, T. Y. Huang, Y. C. Chen, and T. P. Jung, Estimating Driving Performance Based on EEG Spectrum and Fuzzy Neural Network, IEEE International Joint Conference on Neural Networks, (2004).
[22] G. D. Wu and C. T. Lin, A Recurrent Neural Fuzzy Network for Word Boundary Detection in Variable Noise-Level Environments, IEEE Transaction on Systems Man and Cybernetics-Part B: Cybernetics, vol. 31, no. 1, pp. 84-97, (2001).
[23] W. J. Lee, C. S. Sen, and S. J. Lee, Constructing Neuro-Fuzzy Systems with TSK Fuzzy Rules and Hybrid SVD-Based Learning, IEEE International Conference on Fuzzy System, (2002).
[24] C. F. Juang, H. S. Perng, and S. K. Chen, Skin Color Segmentation by Histogram-Based Neural Fuzzy Network, IEEE International Joint Conference on Neural Networks, (2005).
[25] J. C. Bezdek, Pattern recognition with fuzzy objective function algorithms, Plenum Press, New York, (1981).
[26] M. Prasad, C. T. Lin, C. T Yang, and A. Saxena, Vertical Collaborative Fuzzy C-Means for Multiple EEG Data Sets, Springer Lecture Notes in Computer Science, vol. 8102, pp. 246-257, (2013).
[27] K. B. Cho and B. H. Wang, Radial basis function based adaptive fuzzy systems and their applications to system identification and prediction, Fuzzy Sets and Systems, vol. 83, no. 3, pp. 25–339, (1996).
[28] S. Chen, C. F. N. Cowan, and P. M. Grant, Orthogonal least squares learning algorithm for radial basis function network, IEEE Transaction on Neural Networks, vol. 2, no. 14, pp. 11–1423, (1991).
[29] S. Q. Wu and M. J. Er, Dynamic fuzzy neural networks—a novel approach to function approximation, IEEE Transaction and Systems Man and Cybernetics-Part B: Cybernetics, vol. 30, no. 3, pp. 58–364, (2000).
[30] N. Wang, M. J. Er, and X. Y. Meng, A fast and accurate online self-organizing scheme for parsimonious fuzzy neural networks, Neurocomputing, vol. 72, no. 38, pp. 3818–3829, (2009).
[31] J. Platt, A resource-allocating network for function interpolation, Neural Computation, vol. 3, no. 2, pp. 213–225, (1991).
[32] C. S. Velayutham and S. Kumar, Asymmetric Subsethood-Product Fuzzy Neural Inference System (ASuPFuNIS), IEEE Transaction on Neural Networks, vol. 16, no. 1, pp 160–174, (2005).
[33] L. Yingwei, N. Sundararajan, and P. Saratchandran, A sequential learning scheme for function approximation using minimal radial basis function (RBF) neural networks, Neural Computation, vol. 9, no. 4, pp. 461–478, (1997).
[34] G. B. Huang, P. Saratchandran, and N. Sundararajan, A generalized growing and pruning RBF (GGAPRBF) neural network for function approximation, IEEE Transaction on Neural Networks, vol. 16, no. 1, pp. 57–67, (2005).
[35] N. Y. Liang, G. B. Huang, P. Saratchandran, and N Sundararajan, A fast and accurate online sequential learning algorithm for feedforward networks, IEEE Transaction on Neural Networks, vol. 17, no. 6, pp. 1411–1423, (2006).
[36] G. Leng, G. Prasad, and T.M. McGinnity, An On-line Algorithm for Creating Self-organizing Fuzzy Neural Networks, Neural Networks, vol. 17, pp. 1477-1493, (2004).
[37] G. Leng, T. M. McGinnity, and G. Prasad, Design for Self Organizing Fuzzy Neural Network Based on Genetic Algorithm, IEEE Transaction on Fuzzy Systems, vol. 14, no. 6, pp. 755-766, (2006).
[38] O. Khayat, M. M. Ebadzadeh, H. R. Shahdoosti, R. Rajaei, and I. Khajehnasiri, A Novel Hybrid Algorithm for Creating Self-organizing Fuzzy Neural Networks, Neurocomputing, vol. 73, pp. 517-524, (2009).

[39] H. J. Rong, N. Sundararajan, G. B. Huang, and P. Saratchandran, Sequential adaptive fuzzy inference system (SAFIS) for non-linear system identification and prediction, Fuzzy Sets and Systems, vol. 157, no. 9, pp. 1260–1275, (2006).

[40] P. Angelov and D. Filev, An approach to online identification of Takagi–Sugeno fuzzy models, IEEE Transactions on Systems Man and Cybernetics, Part B: Cybernetics, vol. 34, no. 1, pp 484–498, (2004).

[41] H. Malek, M. M. Ebadzadeh, and M. Rahmati, Three New Fuzzy Neural Networks Learning Algorithms Based on Clustering, Training Error and Genetic Algorithm, Springer Science and Business Media, Applied Intelligence, vol. 37, no. 2, pp. 280-289, (2012).

[42] P. Angelov and D. Filev, SIMPL eTS: a simplified method for learning evolving Takagi–Sugeno fuzzy models, IEEE International Conference on Fuzzy Systems, (2005).

[43] W. Klimesch, EEG alpha and theta oscillations reflect cognitive and memory performance: A review and analysis, Brain Research Reviews, vol. 29, nos. 2–3, pp. 169–195, (1999).

[44] S. Hengjie, M. Chunyan, S. Zhiqi, M. Yuan, and B. S. Lee, A fuzzy neural network with fuzzy impact grades, Neurocomputing, vol. 72, nos. 13–15, pp. 3098–3122, (2009).

[45] I. Rojas, H. Pomares, J. L. Bernier, J. Ortega, B. Pino, F. J. Pelayo, and A. Prieto, Time series analysis using normalized PG-RBF network with regression weights, Neurocomputing, vol. 42, nos.1–4, pp. 267–285, (2002).

[46] C. F. Juang and Y. W. Tsao, A self-evolving interval type-2 fuzzy neural network with online structure and parameter learning, IEEE Transaction on Fuzzy Systems, vol. 16, no. 6, pp. 1411– 1424, (2008).

[47] Y. Chen, B. Yang, and J. Dong, Time-series prediction using a local linear wavelet neural network, Neurocomputing, vol. 69, nos. 4–6, pp. 449–465, (2006).

[48] Y. Y. Lin, J. Y. Chang and C. T. Lin, A TSK-type-based Self-Evolving Compensatory Interval Type-2 Fuzzy Neural Network (TSCIT2FNN) and Its Applications, IEEE Transaction on Industrial Electronics, vol. 61, no. 1, (2014).

[49] W. Pedrycz, Collaborative fuzzy clustering, Pattern Recognition Letters, vol. 23, no. 14, pp. 1675–1686, (2002).

[50] W. Pedrycz, Knowledge-based clustering: from data to information granules, A John Wiley &Sons, Inc., Publication, (2005).

[51] C.T. Lin, M. Prasad, and J. Y. Chang, Designing Mamdani Type Fuzzy Rule Using a Collaborative FCM Scheme, International Conference on Fuzzy Theory and Its Application (iFuzzy), Taipei, Taiwan, Dec. 6-8, (2013).

[52] M. Prasad, K. P. Chou, A. Saxena, O. P. Kawrtiya, D. L. Li, and C. T. Lin, Collaborative Fuzzy Rule Learning for Mamdani type Fuzzy Inference System with Mapping of Cluster Centers, IEEE Symposium on Computational Intelligence in Control and Automation (CICA), Orlando, FL, Dec. 9-12, (2014).

[53] K. P. Chou, M. Prasad, Y. Y. Lin, S. Joshi, C. T. Lin, and J. Y. Chang, Takagi-Sugeno-Kang type Collaborative Fuzzy Rule Based System, IEEE symposium on Computational Intelligence and Data Mining (CIDM), Orlando, FL, Dec. 9-12, (2014).

[54] M. Prasad, D. L. Li, Y. T. Liu, L. Siana, C. T. Lin, and A. Saxena, A Preprocessed Induced Partition Matrix Based Collaborative Fuzzy Clustering for Data Analysis, IEEE International Conference of Fuzzy Systems (FUZZ-IEEE), Beijing, China, July 6-11, (2014).