A self-evolving fuzzy system which learns dynamic threshold parameter by itself

DOI:
10.1109/TFUZZ.2018.2886154

Document Version
Accepted author manuscript

A self-evolving fuzzy system which learns dynamic threshold parameter by itself

Dongjiao Ge, *Student Member, IEEE,* Xiao-Jun Zeng *Member, IEEE*

***Abstract*—This paper proposes an online learning algorithm for data streams namely self-evolving fuzzy system (SEFS). Unlike the fixed control parameters commonly used in evolving fuzzy systems, SEFS uses online training errors, which measure the quality of an identified model in presenting the dynamics of the data stream, to set a dynamic threshold automatically for rule generation. This self-tuning parameter, which controls the speed and coverage for fuzzy rule generation, helps SEFS properly deal with the under-fitting/over-fitting problems relying on two facts. (1) Large training errors present an under-fitted model, which is too coarse to represent the highly complicated and rapidly dynamic (e.g. highly nonlinear, non-stationary) behavior of the data segment. Then, finer rules need to be added. (2) Tiny training errors reflect an over-fitted model, which can ideally represent any slight dynamic behavior of the data stream. In this case, coarse rule base should be used. Besides, a L^2 distance based geometric similarity measure is proposed in the rule merging phase. With this similarity measure, SEFS computes the similarity between Gaussian membership functions accurately without making an approximation of the Gaussian membership function beforehand. In addition, a weighted recursive least square algorithm with a variable forgetting factor (VFF-WRLS), which minimizes the mean square of the noise-free posterior error signal, is applied to learn the consequent parameters. Several benchmark examples across both artificial and real-life data sets verify that SEFS has the ability to give better performance compared with many state-of-the-art algorithms.**

***Index Terms*—Evolving fuzzy system, recursive least square, online learning, data stream.**

I. Introduction

STREAMING data are one of the most common types of data in many real world applications. This makes data stream mining a very important research area. A data stream usually comes in high speed, and it is always accompanied with concept drift (also known as non-stationary phenomenon). As a result, data streams bring new challenges for mining techniques in data storage, learning in non-stationary environment as well as quick tracking of the data. Mining data streams involves a lot of research topics such as regression, clustering, classification, summarization and so on [1]. This paper mainly focuses on data stream regression problems, which typically include problems such as time series prediction, function approximation and system identification [2]. With the characteristics of structure and parameters are able to update online, and no requirement for storing huge historical data, evolving fuzzy systems (eFSs) are effective and widely used fuzzy models in dealing with data stream regression issues.

In previous literatures, there are numerous eFSs researches about data stream regression problems. Since the early development of methods for identifying eFSs, there have been two major parts consist of those eFSs identification approaches. These two parts contain antecedent learning (rule generation, rule simplification) and consequent learning (consequent parameters updating). Both of these two blocks severely affect the learning ability of eFSs. Most of existing works focus on studying the first block—making improvement on fuzzy rule generation and simplification criterions, while using the existing optimization approaches directly in consequent learning. Frequently used methods in existing eFSs identification algorithms for antecedent learning could be summarized as follows:

(1) *Fuzzy rule generation:* This process is also known as fuzzy rule adding. Since there is usually no prior knowledge about how many and what fuzzy rules are needed to depict the input space, it is a crucial task to learn the fuzzy rules including the cluster centers and radiuses as well as fuzzy rule numbers online. In the early work of eFSs, a *distance based criterion* has been used. For example, [3] proposed a dynamic evolving neural-fuzzy inference system (DENFIS) adding new fuzzy rules based on the Euclidean distances between the new input data and the existing cluster centers. Furthermore, flexible fuzzy inference systems (FLEXFIS) [4] and dynamically evolving clustering (DEC) [5] judge whether a new input sample is in the existing clusters by comparing distances between this input and the cluster centers with the corresponding radiuses. Considering eFSs developed based on generalized fuzzy rules [6], Mahalanobis distance has been used to control the fuzzy rule growth e.g. [7], [8]. Similar to distance based methods, there exists another effective criterion built according to the activation degree (or firing strength). As activation degree takes the distance between the input and existing cluster centers into consideration by nature, the essence of activation degree is the same as the distance based criterions, but the activation degree is more intuitive to assess whether a data point is close to a cluster center. Approaches such as self-organizing fuzzy neural network (SOFNN) [9], online self-organizing fuzzy modified least-square network (SOFMLS) [10] whose improved approaches could be found from [11], [12], evolving neural-fuzzy semantic memory model (eFSM) [13], and evolving neo-fuzzy neural network approach (eNFN) [14] implement activation

Manuscript received April 23, 2018; revised October 18, 2018; accepted November 22, 2018. The first author, Dongjiao Ge, of this work is funded by the President's Doctoral Scholar Award of the University of Manchester. (*Corresponding author: Xiao-Jun Zeng*).

Dongjiao Ge and Xiao-Jun Zeng are with School of Computer Science, University of Manchester, M13 9PL, Manchester, U.K.(email: dongjiao.ge@manchester.ac.uk; x.zeng@manchester.ac.uk).

degree to make decisions on fuzzy rule adding. Further, recent researches using activation degree, for example, are generic self-evolving Takagi-Sugeno-Kang fuzzy framework (GSETSK) [15], the eFSs identification method proposed by [2], and local error optimization approach (LEOA) [16]. Besides, *datum significance (DS) criterion* is another effective rule generation criterion. As a generalized version of the *significant criterion* [17], [18]–[20] and the *influence* [21], *DS criterion* measures whether the new input could contribute more to the prediction results than existing clusters. Parsimonious network based on fuzzy inference system (PANFIS) [22] and Generic Evolving Neuro-Fuzzy Inference System (GENEFIS) [23] are two typical methods that put forward and apply this criterion. In addition to these approaches for rule generation, there also exist many other criterions, for instance, *rule potential* [24]–[26], *compatibility measure*, [6], [27], *data density* [28].

(2) *Fuzzy rule simplification:* Rule simplification process helps an eFS lower down the computational burden and keep a tidy rule base to make swift predictions. There are two main ways to help eFS eliminate redundant rules. One is fuzzy rule merging and the other is fuzzy rule pruning. Previous researches such as [8], [14], [16], [24], [27], [29], [30] proposed and used rule simplification criterions such as *age*, *utility* to delete clusters which are rarely used. Apart from pruning low utility rules, another desirable approach to simplify the rule base is known as fuzzy rule merging. The process combines two or several similar clusters to one cluster to not only decrease the computational burden, but also avoid the rule conflict. As a result, how to judge the similarity of two fuzzy rules becomes a crucial problem discussed by researches containing merging process. There are two major classes of rule similarity measure: set-theoretic similarity measure and geometric similarity measure [31]. Set-theoretic measures measure the proportion of the intersection of two clusters comparing with the union of these two clusters; while geometric measures are comparing the distance between membership functions of two clusters. Set-theoretic measures are valid measures which can determine the similarity between two overlapping clusters. Examples of previous researches with set-theoretic similarity measures are GSETSK [15], eFSM [13], eT2FIS [32]. Different from set-theoretic measures, geometric measures are distances based effective alternatives. Geometric similarity measures are usually fast and comparatively easy to calculate. Furthermore, they are also widely used in many existing eFSs identification algorithms (e.g. [5]–[7], [9], [16], [22], [23], [27], [28]).

Following the above methods, many different approaches and algorithms for eFSs have been proposed and developed. However, all these approaches still suffer technical limitations from the following two aspects.

- As discussion above, the rule generation of the eFSs could be determined by the distance based criterion, activation degree or other criteria. However, a fundamental problem which has not been effectively solved is how to determine the right threshold value for such a criterion. This threshold value is very crucial in controlling the speed for rule growth and accuracy of the systems. When threshold is set to make the criterion loose, the rule number will growing slowly and the obtained clusters are big ones. If the threshold is set to let the criterion very strict, then it will obtain many tiny clusters. It can be seen from this phenomenon, too loose or too strict criterions are likely to cause under-fitting or over-fitting issues. Unfortunately, the current practice of setting such a control parameter is a fixed value based on the experienced value or trial-error off-line experiment. The experienced value does not work as the different threshold values which are needed when learning different systems, and there is no one value which fits to all the systems. Even setting the individual value for each system to be learned based on the off-line experiment still inappropriate, as data streams always have non-stationary and nonlinear phenomenon. Therefore, a fixed threshold value is hard to generate a fuzzy system with appropriate complexity to approximate the data stream. The reason behind this is that too simple/coarse system is usually lack of ability to fit the highly non-stationary phenomenon, and may cause under-fitting; whereas over complicated system would learn from the noise and lead to over-fitting. Furthermore, a fixed threshold is hard to guarantee that new added rules can ensure the reduction of prediction errors. Therefore, in many applications, it is difficult to find a fixed value threshold, no matter whether it is based on experience or experiment, to make the eFS evolve effectively and accurately according to the state and the need of a data stream.
- Both set-theoretic and geometric similarity measure face two common challenges: i) the direct use of Gaussian membership functions is hard to meet the requirement of the online learning regarding to computation speed [33]; ii) approximations of Gaussian membership functions, and the heuristic similarity measures are difficult to accurately measure the rule similarities. To be more specific, on one aspect, set-theoretic similarity measures are usually computationally expensive when using Gaussian or bell-shaped membership functions [34], because of the difficulty in computing the intersection of fuzzy sets even for the off-line learning. Many alternative methods have been proposed in previous works, such as using triangle [32] or trapezoidal [34] to approximate the Gaussian membership function based on the α-cut of the fuzzy set. These approximated measures are inaccurate due to the different shapes between Gaussian and the approximated membership functions. On the other aspect, geometric measures are distance based measures and easier to compute, as only the distance between the membership functions is required to calculate and so widely used for on-line learning. Considering the computation speed, inaccurate and intuitive approaches (e.g.distances between cluster centers or radiuses, Bhattacharyya distance) have been frequently applied. The assumption behind these measures are that if the parameters or the statistical samples present extremely similar behavior, then the firing strengths would have high similarity. Unfortunately, the existing approaches are approximate or heuristic and so are inaccurate, as a result, they could lead to the wrong merge. Therefore, there is a need to propose a similarity

measure to tackle these problems.

To address these important issues, a self-evolving fuzzy system (SEFS) is proposed in this paper. The main novelties and advantages of SEFS could be summarized as follows.

- Rather than a fixed value, SEFS determines and dynamically tunes the threshold parameter by self-learning from data. Noticing the fact that online training errors can indicate whether the learned eFSs has appropriate complexity to fit the data stream, then a self-learning strategy is proposed. This strategy automatically and dynamically set the threshold parameter to control rule generation based on two basic principles: when the learned eFS is under-fitting, the threshold value is decreased to speed up the rule adding; when the learned eFS is over-fitting, the threshold parameter is increased to slow down the speed of rule adding. In more details, the threshold is set to be a function of the cumulative online training error which is computed by the sum of the output absolute errors with forgetting factor as weights (the older the training errors, the smaller the weights). The small or even tiny online training errors demonstrate that the eFS is complicated enough to fit the data stream, and more complicated system would cause over-fitting. Then, the threshold parameter should be tuned to slow down the speed for generating new rules. Otherwise, the big online training errors illustrate the eFS is not complex enough to catch up the data dynamics, and it is necessary to increase the system complexity. Then, the threshold parameter should be tuned to allow more new rules generated to get rid of under-fitting. As a result, with the time-varying threshold to control the rule growth, SEFS can learn by itself to discover the right value of the threshold parameter; the error based rule generation approach intends to reduce the errors through adjusting the rule adding speed.
- We propose a new geometric similarity measure, which is derived from the idea that two fuzzy rules are similar when they have similar normalized firing strengths everywhere in the domain. This proposed similarity measure has two main novelties: i) an accurate calculation of the similarity is given by the straight forward use of the Gaussian membership functions; ii) with an analytic form, this similarity measure is easy to compute and suitable for online learning. To be more specific, the proposed similarity measure determines the similarity of two fuzzy rules from the difference between the firing strengths, instead of the difference between membership functions in each dimension, and results in an economic rule base without losing the accuracy. Despite of applying triangles or trapezoids to approximate and replace the Gaussian membership functions, the original form of Gaussian membership functions are applied. Besides, unlike the heuristic and indirect approaches to measure the difference of two firing strengths, we measure the L^2 distance between the firing strengths directly in the function space, and induct a easily computed analytic form of this L^2 distance. To summarize, with this accurate similarity measure, SEFS can make fast decisions of rule merging at appropriate occasions on the fly.

The rest of this paper is arranged as follows. Section II presents the basic structure of a evolving fuzzy system and the problem which needs to be solved. Section III proposes and explains the learning details of SEFS in fuzzy rule adding and merging, and antecedent and consequent parameters updating. Numerical examples used to evaluate SEFS are displayed in Section IV. In Section V, conclusions are given.

II. Problem statement

The T-S fuzzy system is considered in this paper. The form of the i-th fuzzy rule R_i is multi-input-single-output (MISO) type shown in (1):

$$R_i: If\ x_1\ is\ \Gamma_{i,1}\ and\ x_2\ is\ \Gamma_{i,2}\ and\ \ldots\ and\ x_n\ is\ \Gamma_{i,n},$$
$$then\ y_i = \psi_{i,0} + \sum_{j=1}^{n} \psi_{i,j} x_j, \tag{1}$$

where $i = 1,2,\ldots,K$, K is the number of fuzzy rules, $x = (x_1,x_2,\ldots,x_n)$, in which x_j is the j-th input, $x \in \Omega = [a_1,b_1] \times [a_2,b_2] \times \ldots \times [a_n,b_n] \subset \mathbb{R}^n$, y_i is the output of rule R_i, $\psi_i = (\psi_{i,0},\psi_{i,1},\ldots,\psi_{i,n})$ is the vector of consequent parameters, n is the number of the input variables.

The membership function of $\Gamma_{i,j}$ is $\mu_{i,j}(x_j)$ which is a Gaussian membership function [12], [19], [20] with form (2),

$$\mu_{i,j}(x_j) = \exp\{-\frac{(x_j - c_{i,j})^2}{2(\sigma_{i,j})^2}\}, \tag{2}$$

in which $c_{i,j}$ and $\sigma_{i,j}$ are cluster center and radius, respectively. Furthermore, for rule R_i, the firing strength $\gamma_i(x)$ and the normalized firing strength $\theta_i(x)$ are presented by (3) and (4), respectively. The final output of the system $\hat{y}$ could be computed by (5).

$$\gamma_i(x) = \prod_{j=1}^{n} \mu_{i,j}(x_j), \tag{3}$$

$$\theta_i(x) = \gamma_i(x) / \sum_{j=1}^{K} \gamma_j(x), \tag{4}$$

and

$$\hat{y} = \sum_{i=1}^{K} \theta_i(x) y_i. \tag{5}$$

This paper is going to propose the one-pass online approach to identify T-S fuzzy system from three aspects: rule number K, antecedent parameters $c_i = (c_{i,1},c_{i,2},\ldots,c_{i,n})$, $\sigma_i = (\sigma_{i,1},\sigma_{i,2},\ldots,\sigma_{i,n})$, as well as consequent parameters $\psi_i = (\psi_{i,0},\psi_{i,1},\ldots,\psi_{i,n})^T$, where $i = 1,2,\ldots,K$.

III. Incremental learning algorithm of SEFS

A. Fuzzy rule adding and updating

Assume that the real input and corresponding output are $x(t) = (x_1(t),x_2(t),\ldots,x_n(t))$ and $y(t)$, and $\hat{y}(t)$ is the output estimated by SEFS. It is the most widely used approach to add a new fuzzy rule if $\forall i = 1,2,\ldots,K$, $\gamma_i(x(t)) < \varepsilon$ holds. Most of the previous researches set ε as a fixed value threshold. Different from the existing approaches, in this paper, this threshold is a time varying and self-tuning threshold ε_t, which

captures the dynamics of the data stream, and adjusts its value by self-learning to follow these dynamics. As the online training errors contain the information of under-fitting and over-fitting of the eFS, the threshold ε_t is designed to be the function of the online training errors. Besides, due to the fact that the most recent data always have higher influence on the future behaviour of the data stream than the old data, it is natural and reasonable that newer training errors should have more influence on the threshold than the older ones. Furthermore, on one hand, when the cumulative online training error is big, the learned eFS is too coarse to catch up the non-linearity and non-stationary of the data stream, then, more fuzzy rules are required to overcome the under-fitting problem. In this case, a big threshold ε_t enables the eFS to evolve rapidly to increase the accuracy. On the other hand, a small (or tiny) cumulative online training errors demonstrate that the eFS is complex (or over complicated) to approximate the data stream. In this situation, increasing the rule numbers in a high speed is likely to lead to over-fitting, thus, a small threshold ε_t is more appropriate for slowing down the rule adding speed to avoid over-fitting. As a result, in order to preciously depict the phenomenon of the variation of the threshold along with the ability of eFS to fit the dynamics of the data stream, the threshold ε_t (6) is designed as the monotonic increasing function of the cumulative online training error $\sum_{k=1}^{t} \lambda^{t-k} e_k$. Because the direct use of the training errors $\hat{y}_k - y_k$ to compute the cumulative error will lead to the positive and negative values compensate each other, therefore, the absolute training errors $e_k = |\hat{y}_k - y_k|$ are applied.

$$\varepsilon_t = \varepsilon_{max} - (\varepsilon_{max} - \varepsilon_{min}) E_t, \text{ where} \tag{6}$$

$$E_t = \exp\{-\sum_{k=1}^{t} \lambda^{t-k} e_k\}, \tag{7}$$

$\lambda \in [\lambda_0, 1)$ is the forgetting factor for indicating the importance of each error, $e_k = |y(k) - \hat{y}(k)|$, which is the absolute error, and the interval $[\lambda_0, 1)$ is the admissible forgetting factor interval. Smaller λ permits faster forgetting of the old errors and more focus on information contained in the recent errors. The lower and upper bounds of the threshold ε_t are ε_{min} and ε_{max}, respectively. Assume $E_{t-1} = \exp\{-A(t-1)\}$ and $A(t-1) = \sum_{k=1}^{t-1} \lambda^{t-1-k} e_k$, then $A(t)$ could be updated by $A(t) = \lambda A(t-1) + e_t$.

Proposition III.1. *ε_t is monotonically increasing against $\sum_{k=1}^{t} \lambda^{t-k} e_k$.*

Proof. See appendix A. □

Definition III.1. *(ε-completeness [35]) For any input $x \in \Omega$, there exists at least one fuzzy rule such that the activation degree of x is no less than ε.*

Now the *rule adding method* of SEFS is proposed as below.

Rule adding method. *If $\gamma_i(x(t)) < \varepsilon_t$, $\forall\ i = 1, 2, \ldots, K$, then add a new fuzzy rule with cluster center $x(t)$. Radius is $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_n)$ with $\sigma_j = \frac{\|c_{i^*} - x(t)\|_2}{k_0}$ ($k_0 = \sqrt{-2\log(\varepsilon_t)}$, $j = 1, 2, \ldots, n$), in which $i^* = \arg\max_i \gamma_i(x(t))$. ε_t is a time varying threshold with the function form shown in (6).*[1]

Rule adding method helps to determine the radius of the new cluster by assuring c_{i^*}, which is the nearest cluster center of $x(t)$, satisfies that $\exp\{-\sum_{j=1}^{n} \frac{(c_{i^*,j} - x_j(t))^2}{2\sigma_j^2}\} = \varepsilon_t$. According to the bounds of ε_t, the corresponding dynamic parameter $k_0 = k_t = \sqrt{-2\log(\varepsilon_t)}$ is in the interval $[\sqrt{-2\log(\varepsilon_{max})}, \sqrt{-2\log(\varepsilon_{min})}]$. The form of k_t in *rule adding method* is the generalized form of which in [22], [23], [36].

The radius setting method given by *rule adding method* is applicable when $K > 0$. However, at the very beginning of SEFS learning from the data, the first input $x(1)$ is set as the center of the first cluster and the radius could not be set by *rule adding method*. In order to deal with this, we set the original radius of the first cluster as 0, and when the second input $x(2)$ comes, a new cluster is built with center $x(2)$, and the radiuses of the first and second cluster are set as $\sigma_{1,j} = \sigma_{2,j} = \frac{1}{k_1}\|c_1 - c_2\|_2$ ($k_1 = \sqrt{-2\log(\varepsilon_1)}$), $j = 1, 2, \ldots, n$. If *rule adding method* is not satisfied, then SEFS updates the cluster center and radius of rule R_{i^*} according to *rule updating method*, the formulas in which are based on the sample mean $\hat{c}$ and variance $\hat{\sigma}^2$ shown in (8),

$$\hat{c} = \frac{\sum_{k=1}^{N} x(k)}{N},\ \hat{\sigma}^2 = \frac{\sum_{k=1}^{N} (x(k) - \hat{c})^2}{N}, \tag{8}$$

where $x(1), x(2), \ldots, x(N)$ are the samples, N is the number of samples.

Rule updating method. *If $\gamma_{i^*}(x(t)) \geq \varepsilon_t$, then c_{i^*} and σ_{i^*} should be updated by the following recursive formulas*

$$c_{i^*,j}(t) = c_{i^*,j}(t-1) + \frac{x_j(t) - c_{i^*,j}(t-1)}{N_{i^*}(t-1) + 1}, \tag{9}$$

$$\begin{aligned}(\sigma_{i^*,j}(t))^2 =& (\sigma_{i^*,j}(t-1))^2 + \frac{(x_j(t) - c_{i^*,j}(t))^2 - (\sigma_{i^*,j}(t-1))^2}{N_{i^*}(t-1) + 1} \\ &+ \frac{N_{i^*}(t-1)(c_{i^*,j}(t) - c_{i^*,j}(t-1))^2}{N_{i^*}(t-1) + 1},\end{aligned} \tag{10}$$

where N_{i^*} is the number of inputs with firing strengths larger than ε_t, $j = 1, 2, \ldots, n$.[2]

Proof. The induction of formula (9) and (10) is presented in the appendix B. □

Existing methods which use the same approach for rule updating are, for example, DENFIS [3], FLEXFIS [4].

B. Fuzzy rule merging

Fuzzy rules with similar antecedent parts but different consequent parts are very likely to cause rule confliction. Merging fuzzy rules with similar antecedent parts is an effective approach to avoid rule confliction and redundancy.

[1] k_0 is obtained by $\varepsilon_t = \exp\{-(k_0)^2/2\}$. $\|\cdot\|_2$ is the Euclidean norm.

[2] N_i for the rule R_i is regarded as the number of inputs which determine the real position and shape of the corresponding cluster of rule R_i. N_i should be recorded from the time that the rule R_i is built. Once *rule updating method* holds for the rule R_{i^*}, then $N_{i^*} = N_{i^*} + 1$.

More importantly, in the context of online learning, rule adding without merging will lead to the rule number keeping increase and result in an unnecessarily too complicated fuzzy system. Therefore, it is crucial to make judgement of the similarity between the fuzzy rules and then merge the similar rules. As summarized in the introduction, there are many existing works on both set-theoretic and geometric similarity measures. Set-theoretic similarity measures require to compute the intersection and the union of the fuzzy sets. Due to the non-linearity of the Gaussian membership function, it is difficult to compute the set-theoretic similarity measures [34], [37]. Therefore, the commonly used approach is to use the piece-wise linear approximation of the Gaussian membership function, and two typical examples to make this approximation are to use triangle [37] or trapezoidal [34] membership functions. There are a small number of researches in recent years, for instance, [38], attempted to compute the intersection and union of the fuzzy sets using Gaussian membership function directly, without making any approximations in advance. However, this method is computationally expensive, and under the risk of the curse of dimensionality. Comparing with set-theoretic similarity measures, geometric measures are easier to compute and widely used in many existing works of eFS. The most widely used geometric measure is based on the distance between the parameters of the membership functions, and the examples could be found from [23], [22], [28], [27]. However, it is still hard to make the accurate judgement for whether the distance between the firing strengths is small from only comparing the distance between the antecedent parameters. As a result, no matter set-theoretic or geometric similarity measures cannot measure the distance between the firing strengths (or Gaussian membership functions) directly and accurately. In addition, to our best knowledge, there is no analytic form to compute the distance between the firing strengths computed using Gaussian membership functions presented in the existing researches.

In SEFS, an analytic form of the similarity measure between the firing strengths based on L^2 distance is proposed, which is a direct and accurate method of computing the similarity rather than the indirect and approximate ones in the literature. Further, our new similarity measure could be worked out very fast. Follow from the formal definition of the geometric similarity measure presented in [31], the similarity measure for rule R_1 and R_2 could be presented by $S(R_1,R_2)=\frac{1}{1+D(R_1,R_2)}$, where $D(R_1,R_2)$ is the distance between rule R_1 and R_2. Set $D(\cdot,\cdot)$ be the L^2 distance, then *definition III.2* could be obtained.

Definition III.2. *Assume that γ_{i_1} and γ_{i_2} are the firing strengths of rules R_{i_1} and R_{i_2}, respectively. The similarity between these two fuzzy rules is defined as $S(R_{i_1},R_{i_2})$ presented by (11),*

$$S(R_{i_1},R_{i_2})=\frac{1}{1+\|\gamma_{i_1}-\gamma_{i_2}\|_{L^2}}, \tag{11}$$

where $\|\cdot\|_{L^2}$ is the L^2 norm.

The following part of this section will develop the method on how to compute the L^2 distance $\|\gamma_{i_1}-\gamma_{i_2}\|_{L^2}$ fast and precisely.

The L^2 distance in (11) could be represented by formula (12),

$$\begin{aligned}\|\gamma_{i_1}-\gamma_{i_2}\|_{L^2}^2 &= \int_\Omega |\gamma_{i_1}(x)-\gamma_{i_2}(x)|^2\mathrm{d}x\\ &= \int_{a_n}^{b_n}\cdots\int_{a_1}^{b_1}[\prod_{j=1}^{n}\mu_{i_1 j}(x_j)-\prod_{j=1}^{n}\mu_{i_2 j}(x_j)]^2\mathrm{d}x_1\cdots\mathrm{d}x_n\\ &= \prod_{j=1}^{n}\int_{a_j}^{b_j}[\mu_{i_1,j}(x_j)]^2\mathrm{d}x_j+\prod_{j=1}^{n}\int_{a_j}^{b_j}[\mu_{i_2,j}(x_j)]^2\mathrm{d}x_j\\ &\quad -2\prod_{j=1}^{n}\int_{a_j}^{b_j}[\mu_{i_1,j}(x_j)\mu_{i_2,j}(x_j)]\mathrm{d}x_j,\end{aligned} \tag{12}$$

$\mu_{i_1,j}$ and $\mu_{i_2,j}$ are both Gaussian membership functions defined by (2). In the following parts we calculate those three terms in (12) separately.

Let $t_j=\frac{x_j-c_{i_1,j}}{\sigma_{i_1,j}}$, $a_j=\sigma_{i_1,j}a_{i_1,j}+c_{i_1,j}$, $b_j=\sigma_{i_1,j}b_{i_1,j}+c_{i_1,j}$, then we have

$$\begin{aligned}\prod_{j=1}^{n}\int_{a_j}^{b_j}[\mu_{i_1,j}(x_j)]^2\mathrm{d}x_j &= \prod_{j=1}^{n}\int_{a_j}^{b_j}\exp\{-(\frac{x_j-c_{i_1,j}}{\sigma_{i_1,j}})^2\}\mathrm{d}x_j\\ &= \prod_{j=1}^{n}\sigma_{i_1,j}\int_{a_{i_1,j}}^{b_{i_1,j}}\exp\{-t_j^2\}\mathrm{d}t_j\\ &= (\frac{1}{2}\sqrt{\pi})^n\prod_{j=1}^{n}\sigma_{i_1,j}G(a_{i_1,j},b_{i_1,j}).\end{aligned} \tag{13}$$

Function $G(a,b)$ has the form shown in (14),

$$G(a,b)=\begin{cases}erf(b)-erf(a) & 0\le a\le b\\ erf(-a)+erf(b) & a\le 0\le b\\ erf(-a)-erf(-b) & a\le b\le 0,\end{cases} \tag{14}$$

where $erf(\cdot)$ is the error function expressed as (15),

$$erf(x)=\frac{1}{\sqrt{\pi}}\int_{-x}^{x}\exp\{-t^2\}dt=\frac{2}{\sqrt{\pi}}\int_{0}^{x}\exp\{-t^2\}dt. \tag{15}$$

Error function could be calculated by some very simple formulas with high accuracy. Two simple examples taken from [39] to compute $erf(\cdot)$ are listed in (16) and (17),

$$erf(x)\approx 1-\frac{1}{(1+a_1x+a_2x^2+a_3x^3+a_4x^4)^4}, \tag{16}$$

where $a_1=0.278393$, $a_2=0.230389$, $a_3=0.000972$, $a_4=0.078108$,

$$erf(x)\approx 1-(a_1t+a_2t^2+a_3t^3+a_4t^4+a_5t^5)\exp\{-x^2\}, \tag{17}$$

in which $t=\frac{1}{1+px}$, $p=0.3275911$, $a_1=0.254829592$, $a_2=-0.284496736$, $a_3=1.421413741$, $a_4=-1.453152027$, $a_5=1.061405429$. The maximum error obtained by (16) and (17) are 5×10^{-4} and 1.5×10^{-7}, respectively. In this paper, the MATLAB built-in function $erf(x)$, which is a rational function approximation shown in [40], is used for approximating the error function. Similar to the first term of (12), the second term of (12) could be computed by (18),

$$\prod_{j=1}^{n}\int_{a_j}^{b_j}[\mu_{i_2,j}(x_j)]^2\mathrm{d}x_j=(\frac{1}{2}\sqrt{\pi})^n\prod_{j=1}^{n}\sigma_{i_2,j}G(a_{i_2,j},b_{i_2,j}). \tag{18}$$

Let $t_j = \{x_j - \frac{\sigma_{i_1,j}^2 c_{i_2,j} + \sigma_{i_2,j}^2 c_{i_1,j}}{\sigma_{i_1,j}^2 + \sigma_{i_2,j}^2}\} / \{\sqrt{2}\frac{\sigma_{i_1,j}\sigma_{i_2,j}}{\sqrt{\sigma_{i_1,j}^2 + \sigma_{i_2,j}^2}}\}$, formula (19) gives how to compute the third term in (12),

$$\begin{aligned}
&-2\prod_{j=1}^{n}\int_{a_j}^{b_j}[\mu_{i_1,j}(x_j)\mu_{i_2,j}(x_j)]\mathrm{d}x_j \\
&=-2(\frac{1}{2}\sqrt{\pi})^n\prod_{j=1}^{n}\frac{\sqrt{2}\sigma_{i_1,j}\sigma_{i_2,j}}{\sqrt{\sigma_{i_1,j}^2+\sigma_{i_2,j}^2}}\exp\{-\frac{(c_{i_1,j}-c_{i_2,j})^2}{2(\sigma_{i_1,j}^2+\sigma_{i_2,j}^2)}\} \\
&\cdot G(a_{i_{1,2},j}, b_{i_{1,2},j}),
\end{aligned} \tag{19}$$

where $a_j = \frac{\sigma_{i_1,j}^2 c_{i_2,j} + \sigma_{i_2,j}^2 c_{i_1,j}}{\sigma_{i_1,j}^2 + \sigma_{i_2,j}^2} + a_{i_{1,2},j}\frac{\sqrt{2}\sigma_{i_1,j}\sigma_{i_2,j}}{\sqrt{\sigma_{i_1,j}^2+\sigma_{i_2,j}^2}}$, $b_j = \frac{c_{i_1,j}\sigma_{i_2,j}^2 + c_{i_2,j}\sigma_{i_1,j}^2}{\sigma_{i_1,j}^2+\sigma_{i_2,j}^2} + b_{i_{1,2},j}\frac{\sqrt{2}\sigma_{i_1,j}\sigma_{i_2,j}}{\sqrt{\sigma_{i_1,j}^2+\sigma_{i_2,j}^2}}$.

Apply (13), (18) and (19), it can be summarized that (12) can be computed by (20),

$$\begin{aligned}
&\int_\Omega |\gamma_{i_1}(x) - \gamma_{i_2}(x)|^2 \mathrm{d}x \\
&=(\frac{1}{2}\sqrt{\pi})^n\{\prod_{j=1}^{n}\sigma_{i_1,j}G(a_{i_1,j},b_{i_1,j}) + \prod_{j=1}^{n}\sigma_{i_2,j}G(a_{i_2,j},b_{i_2,j}) \\
&-2\prod_{j=1}^{n}\frac{\sqrt{2}\sigma_{i_1,j}\sigma_{i_2,j}}{\sqrt{\sigma_{i_1,j}^2+\sigma_{i_2,j}^2}}\exp\{-\frac{(c_{i_1,j}-c_{i_2,j})^2}{2(\sigma_{i_1,j}^2+\sigma_{i_2,j}^2)}\}G(a_{i_{1,2},j},b_{i_{1,2},j})\}
\end{aligned} \tag{20}$$

It could be seen from (20), the computation speed is decided by $\exp(x)$ function, which is caused by the rational function approximation of the $erf(x)$. This rational function approximation contains the multiplication of a polynomial of the 8th (or lower than 8th) degree and an exponential function, and achieves maximal relative errors ranging down to between 6×10^{-19} and 3×10^{-20} [40]. Because it is faster to compute $exp(x)$ for just a few times than a vast amount of times required by the multiple numerical integral, as a result, it is realistic and appropriate to use (20) to compute the similarity $S(R_{i_1}, R_{i_2})$ in online computing. Based on (20), the fuzzy rule merging method of SEFS is constructed in *rule merging method* below.

Rule merging method. *If rule R_{i^*} is updated following the* rule updating method*, and there exist rule $R_{i_1}, R_{i_2}, \ldots, R_{i_M}$ such that $S(R_{i^*}, R_{i_k}) > \delta$ ($k = 1,2,\ldots,M$), then merge rule R_{i^*} with $R_{i_{k^*}}$ ($i_{k^*} = \arg\max_{i_k} S(R_{i^*}, R_{i_k})$) to one rule $R_{i'}$. The center and radius of rule $R_{i'}$ are $c_{i'}$ and $\sigma_{i'} = (\sigma_{i',1}, \sigma_{i',2}, \ldots, \sigma_{i',n})$ calculated by (21) and (22), respectively. The consequent parameters of rule $R_{i'}$ could be computed by $\psi_{i'} = \frac{N_{i^*}\psi_{i^*} + N_{i_{k^*}}\psi_{i_{k^*}}}{N_{i^*}+N_{i_{k^*}}}$. Remove the original two rules R_{i^*} and $R_{i_{k^*}}$, and then set rule $R_{i'}$ to be rule R_{i^*}. (Each time when two rules are merged, then the number of rules K should be updated by $K-1$.) Repeat the above whole process until there is no fuzzy rule which has the similarity degree with rule i^* larger than the threshold.*

$$c_{i'} = \frac{N_{i^*}c_{i^*} + N_{i_{k^*}}c_{i_{k^*}}}{N_{i^*}N_{i_{k^*}}}, \tag{21}$$

$$\begin{aligned}
(\sigma_{i',j})^2 =& \frac{N_{i^*}\sigma_{i^*,j}^2 + N_{i_{k^*}}\sigma_{i_{k^*},j}^2 + N_{i^*}(c_{i'}-c_{i^*})^2}{N_{i^*}+N_{i_{k^*}}} \\
&+ \frac{N_{i_{k^*}}(c_{i'}-c_{i_{k^*}})^2}{N_{i^*}+N_{i_{k^*}}},
\end{aligned} \tag{22}$$

where N_{i^*} and $N_{i_{k^*}}$ are defined the same as the rule updating method*, and $j = 1,2,\ldots,n$.*

Rule merging method illuminates that, once there is a rule R_{i^*} is updated and its center and radius are changed, then there is a need to consider whether it evolves similar to other rules. It is likely that there are more than one fuzzy rule have the similarity degree with rule R_{i^*} larger than the threshold. In this case, it is plausible to merge R_{i^*} with the rule which has the largest similarity degree with rule R_{i^*}. If the fuzzy rule which has the largest similarity degree with rule R_{i^*} is not unique, then merge the earliest built rule with the rule R_{i^*}. The formulas for computing the new cluster center and radius are displayed by (21) and (22), respectively. Similar methodology could also be found from [41]. The induction of formulas (21) and (22) is presented in the appendix B.

Besides, when a evolving fuzzy system evolves as *rule adding method*, *rule updating method* and *rule merging method*, then this fuzzy system satisfies the ε-completeness property.

Theorem III.1. *(ε-completeness of SEFS) Assume that the input space Ω is a bounded and closed subset of $\mathbb{R}^n$.*

(1) Rule adding: *Assume that the existing cluster centers and radiuses are c_i and σ_i with $i = 1,2,\ldots,N_0'$. Furthermore, assume that fuzzy rules are added according to* rule adding method*, then $\forall x \in \Omega$, $\exists i$ and $K < \infty$ ($i \in \{1,2,\ldots,K\}$) such that the firing strength of x satisfies $\gamma_i(x) \geq \varepsilon_{min}$.*

(2) Rule updating: *Assume $\exists K < \infty$ such that $\forall x \in \Omega$, $\exists i \in \{1,2,\ldots,K\}$ with $\gamma_i(x) \geq \varepsilon_{min}$. Further, assume the i^*-th rule is updated by $x(t+1)$. Then, for all $x \in B(c_{i^*}(t), k_{min}\sigma_{i^*}(t))$, it could be inducted that $x \in B(c_{i^*}(t+1), k_{min}\sigma_{i^*}(t+1))$, $k_{min} = \sqrt{-2\log\varepsilon_{min}}$ when $N_{i^*} \to \infty$.*

(3) Rule merging: *Assume that $\exists K < \infty$ such that $\forall x \in \Omega$, $\exists i \in \{1,2,\ldots,K\}$ with $\gamma_i(x) \geq \varepsilon_{min}$. Further, assume $\exists i_1, i_2 \in \{1,2,\ldots,K\}$ such that $\| \gamma_{i_1}(x) - \gamma_{i_2}(x) \|_{L^2} < \varepsilon$, and these two rules are merged to one rule i_0. Then, for all $x \in B(c_{i_1}(t), k_{min}\sigma_{i_1}(t)) \cup B(c_{i_2}(t), k_{min}\sigma_{i_2}(t))$, it could be inducted that $x \in B(c_{i_0}(t+1), k_{min}\sigma_{i_0}(t+1))$ when $N_{i_1}, N_{i_2} \to \infty$.*

Proof. Proof could be found from the appendix D. □

C. Consequents learning

Consequent parameters are updated by the recursive least square algorithm with variable forgetting factor (VFF-RLS) proposed by [42]. VFF-RLS is an improved version of the recursive least square (RLS) method with better performance on tracking sudden system changes. VFF-RLS has an optimal dynamic forgetting factor obtained from minimizing the mean-square noise-free posterior error, and usually has better performance in both stationary and non-stationary environment [42]. In this paper, we generalize VFF-RLS to a local optimum version known as the weighted recursive least square algorithm

with variable forgetting factor (VFF-WRLS) to minimize the error functions (23),

$$Err_i = \sum_{k=1}^{t} \theta_i(k)\lambda_i^{t-k}(y(k) - xe(k)\psi_i)^2, \tag{23}$$

where $xe(k) = (1, x(k))$, $\theta_i(k) = \theta_i(x(k))$, $i = 1, 2, \ldots, K$. Assume that the new input is $x(t)$, and the corresponding output is $y(t)$. The recursive least square updating formulas[3] of ψ_i are

$$\psi_i(t) = \psi_i(t-1) + \frac{\theta_i(t)P_i(t-1)xe(t)(y(t) - \hat{y}(t))}{\lambda_i(t) + \theta_i(t)xe(t)^T P_i(t-1)xe(t)}, \tag{24}$$

$$P_i(t) = \frac{1}{\lambda_i(t)}(P_i(t-1) - \frac{\theta_i(t)P_i(t-1)xe(t)xe(t)^T P_i(t-1)}{\lambda_i(t) + \theta_i(t)xe(t)^T P_i(t-1)xe(t)}), \tag{25}$$

in which $\lambda_i(t)$ is the time varying forgetting factor. Following [42], the initial value of $P_i(t)$ is set as a big unit matrix $P_i(0) = M_0 * I_{(n+1)\times(n+1)}$ (M_0 is a big number). Further, $\lambda_i(t)$ is given in (26), which is the optimum forgetting factor obtained by minimizing $E[(\varepsilon_i'(t))^2]$ where $\varepsilon_i'(t) = \theta_i(t)[xe(t)h(t-1) - xe(t)\psi_i(t)]$ and $h(t-1)$ are the noise-free posterior error and the impulse response, respectively. Further, according to [42], the optimum $\lambda_i(t)$ could be computed by (26),

$$\lambda_i(t) \approx 1 - \frac{2E[e_i'(t)^2]}{M(E[e_i'(t)^2] + \sigma_v^2)}, \tag{26}$$

where M is a big value, $E[e_i'(t)^2]$ is the excess mean square error (EMSE), $e_i(t) = \theta_i(t)[y(t) - xe(t)\psi_i(t-1)] = e_i'(t) + \theta_i(t)v_t$ is the prior error signal, $y(t) = xe(t)h(t-1) + v_t$, $e_i'(t) = \theta_i(t)[xe(t)h(t-1) - xe(t)\psi_i(t-1)]$, σ_v^2 is the variance of Gaussian noise signal v_t. The noise-free prior error signal $e_i'(t)$ is estimated by the *non-iterative shrinkage* method. This method has been used in [43] and [44] to deal with the image de-noising problem. This method recovers a noise-free signal ρ_f from a noisy signal $\rho = \rho_f + \iota$, in which ι is a white Gaussian noise signal, from solving the following $l_1 - l_2$ minimization problem (27),

$$\min_{\rho_f} \ \alpha\|\rho_f\|_1 + 0.5\|\Lambda\rho_f - \rho\|_2^2. \tag{27}$$

In (27), α is the threshold, and Λ is an orthogonal matrix. In the VFF-RLS algorithm, $\rho_f = e_i'(t)$, $\rho = e_i(t)$, $\Lambda = 1$. [42] has shown the optimal solution of (27) could be presented by (28).

$$e_i'(t) = \text{sign}\{e_i(t)\}\max\{|e_i(t)| - \alpha,\ 0\}. \tag{28}$$

The excess mean square error (EMSE) $E[e_i'(t)^2]$ is estimated by the time average of $(e_i'(t))^2$, and presented by (29),

$$E[e_i'(t)^2] = \beta_2 E[e_i'(t-1)^2] + (1-\beta_2)(e_i'(t))^2. \tag{29}$$

The threshold parameter α is taken as the time average of the variance of the white noise v_t. That is $\alpha = \sqrt{\beta_1\sigma_v^2}$. As $y(t) = xe(t)h(t-1) + v_t$, then the variance of v_t could be updated by (30),

$$\sigma_v^2 = \frac{t-1}{t}\sigma_v^2 + \frac{e_i(t)^2}{t}. \tag{30}$$

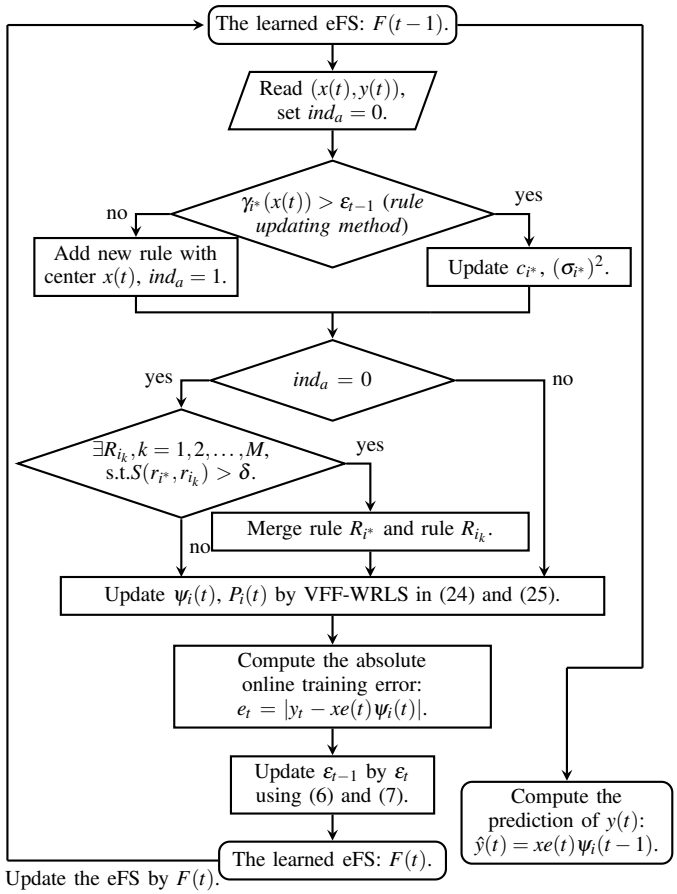


Fig. 1. The flowchart of SEFS.

D. The algorithm: SEFS

According to the antecedent, consequent and rule number learning methods in section III, the flowchart of SEFS algorithm is shown in Fig. 1. Ind_a is the indicator for fuzzy rule adding. The default value of ind_a is 0. If a fuzzy rule is added, then $ind_a = 1$. Further, Fig. 1 presents the online learning and updating framework of SEFS when $(x(t), y(t))$ comes.

IV. Numerical results

Benchmark examples across nonlinear system identification, Mackey-Glass chaotic time series prediction, and real non-stationary time series prediction are shown in this section. These examples are applied to demonstrate that SEFS can effectively solve online regression problems. In these examples, SEFS is running in an online mode. The numerical results are evaluated by the rooted mean square errors (RMSEs(31)) (e.g. [3], [12], [15], [19], [20], [24]), and the non-dimensional error indexes (NDEIs (32)) (e.g. [3], [5], [15], [22], [23], [26]).

$$RMSE = \frac{1}{t}\sum_{k=1}^{t}(y(k) - \hat{y}(k))^2, \tag{31}$$

$$NDEI = \sqrt{RMSE/std(y)}. \tag{32}$$

[3]The induction of (24) and (25) is shown in the appendix E.

TABLE I
SENSITIVITY ANALYSIS (EXAMPLE 1 IN SECTION IV-B).

$\varepsilon_{\max}$	0.5	0.55	0.6	0.65	0.7	s=0.2600
No. of rules	6	6	6	7	6	
RMSE	0.0474	0.0452	0.0428	0.0439	0.0506	
NDEI	0.0459	0.0438	0.0414	0.0425	0.0490	
$\varepsilon_{\min}$	0.4	0.45	0.5	0.55	0.6	s=0.2744
No. of rules	6	6	6	6	6	
RMSE	0.0476	0.0458	0.0428	0.0461	0.0511	
NDEI	0.0461	0.0444	0.0414	0.0447	0.0494	
δ	0.5	0.6	0.7	0.8	0.9	s=0.3718
No. of rules	4	4	6	8	16	
RMSE	0.0500	0.0523	0.0428	0.0483	0.0487	
NDEI	0.0484	0.0507	0.0414	0.0467	0.0471	

A. Sensitivity analysis

Control parameters $\varepsilon_{\max}$, $\varepsilon_{\min}$, δ ($\varepsilon_{max} \geq \varepsilon_{min}$) need to be identified beforehand. Inspired by [45]–[47] the sensitivity s of each parameter ξ_i (i=1,2,3,4 in this paper) is computed by (33):

$$s = \frac{1}{\kappa(\pi_{\max} - \pi_{\min})} \sum_{j=1}^{\kappa-1} |\text{RMSE}(\xi_i^{(j+1)}) - \text{RMSE}(\xi_i^{(j)})|, \tag{33}$$

where κ is the number of samples for a particular parameter, RMSE(ξ_i) is the RMSE computed when using a particular parameter ξ_i, $\pi_{\max}$ and $\pi_{\min}$ are the upper and lower bound of RMSEs. Parameters should be in $[0,1]$ and neither be too big nor too small. In this example, $\pi_{\max}$ and $\pi_{\min}$ are set as the maximum and the minimum value of RMSEs obtained from all the different setting of parameters. In this example, $\pi_{\max}$ and $\pi_{\min}$ are 0.0523 and 0.0428, respectively. This example allows rule generation control parameters $\varepsilon_{\max}$ and $\varepsilon_{\min}$ gear at [0.5, 0.55, 0.6, 0.65, 0.7] and [0.4, 0.45, 0.5, 0.55, 0.6], respectively. The parameter δ that controls the rule merging is varying from [0.5, 0.6, 0.7, 0.8, 0.9]. When each of these parameters are varying from their domains, other parameters are kept as $\varepsilon_{\max} = 0.6$, $\varepsilon_{\min} = 0.5$, $\delta = 0.7$.

Because the nonlinear dynamic plant data in example 1 section IV-B has obvious concept drift based on the 3-state-shift, the data generated from (34) and (35) are applied to analyse the sensitivity of the predefined parameters. We use all 3000 data pairs for evaluation.

It can be seen from Table I, δ is a little bit more sensitive than $\varepsilon_{\max}$ and $\varepsilon_{\min}$. When $\delta < 0.7$ the fuzzy system uses 4 rules to track the data stream and get worse accuracy; and when $\delta > 0.7$ the fuzzy system applies more than 8 rules to track the data stream and get worse accuracy than $\delta = 0.7$. The reason behind this phenomenon is the under-fitting and the over-fitting problem. Besides, $\varepsilon_{\min}$ and $\varepsilon_{\max}$ suffer the similar problems.

Therefore, in all the numerical examples (section IV-B — IV-D), $\varepsilon_{\max} = 0.6$, $\varepsilon_{\min} = 0.5$, $\delta = 0.7$ are kept the same. Based on the existing research, we use the forgetting factor $\lambda = 0.9$. Besides, according to [42] the parameters in VFF-WRLS are $\beta_1 = 8$, $\beta_2 = 0.9$, $M_0 = 10^4$, $M = 10^3$.

TABLE II
EXAMPLE 1: NONLINEAR DYNAMIC PLANT IDENTIFICATION WITH TIME VARYING CHARACTERISTIC.

	No. of rules (AVG.)	No. of parameters (AVG.)	RMSE
SEFS	6(3.5610)	63(24.9270)	0.0428
GSETSK [15]	11	77	0.0661
DENFIS [3]	15	105	0.1749
eTS [24]	11	77	0.0682

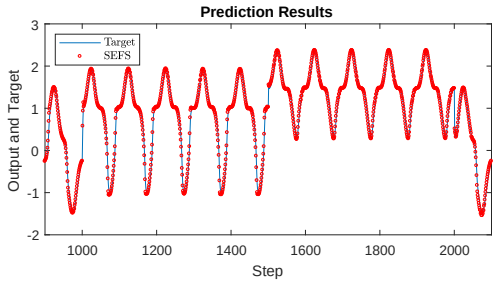


Fig. 2. Online identification results in example 1. ($t \in [900, 2100]$)

B. Example 1: Nonlinear dynamic plant identification with time varying characteristics

The same as [15], the model is (34),

$$y(t+1) = \frac{y(t)}{1+y^2(t)} + u^3(t) + n(t), \tag{34}$$

where $u(t) = \sin(2\pi t/100)$, $n(t)$ is a time-varying factor which could be presented by (35),

$$n(t) = \begin{cases} 0 & 1 \leq t \leq 1000 \text{ and } t \geq 2001 \\ 0.5 & 1001 \leq t \leq 1500 \\ 1 & 1501 \leq t \leq 2000 \end{cases} \tag{35}$$

There are 3000 data points with $t \in [1, 3000]$ generated, and all of them are used for online learning and testing. $(u(t), y(t))$ is the input, and $y(t+1)$ is the output. Numerical results are shown in Table II. Fig. 2 displays the results from $t = 900$ to $t = 2100$, which contains the time of the two drifts at $t = 1000$ and $t = 1500$. Fig. 2 indicates SEFS can adapt to the changes of the state successfully with high accuracy. The running time for SEFS is 0.416035s calculated by a intel(R) core (TM) i7-4790CPU with a 3.6 GHz processor and 16.0 GB memory.

C. Example 2: Mackey-Glass Chaotic time series (long term prediction)

Mackey-Glass Chaotic time series prediction example is a widely used benchmark example in existing works such as [5], [15], [23], [24], [28], [30]. Data is generated from the system (36),

$$\frac{d[x(t)]}{dt} = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t), \tag{36}$$

where $\tau = 17$, $x(0) = 1.2$. There are totally 6000 observations generated. 3000 data points from $201 \leq t \leq 3200$ are used for

TABLE III
EXAMPLE 2: MACKEY-GLASS CHAOTIC TIME SERIES (LONG TERM PREDICTION).

	No. of rules (AVG.)	No. of parameters (AVG.)	NDEI
SEFS	4(1.3043)	52(16.9559)	0.1287
DENFIS [3]	58	886	0.278
eTS+ [26]	10	130	0.392
Simple_eTS+ [29]	15	150	0.375
GENEFIS(C) [23]	19	475	0.280
GENEFIS(B) [23]	9	225	0.339
PANFIS [22]	19	475	0.301
eT2RFNN [28]	3	108	0.32
GSETSK [15]	19	247	0.347
SPLAFIS [30]	30	390	0.279
DeTS [5]	3	39	0.440
GEFNS [48]	5	65	0.2635

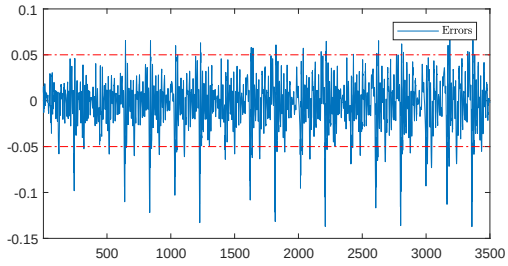


Fig. 3. Online prediction errors of all 3500 data points in example 2.

online training, and 500 data points ranging from $5001 \leq t \leq 5500$ are used for testing. The prediction model is of the form

$$\hat{x}(t+85) = f(x(t-18), x(t-12), x(t-6), x(t)). \quad (37)$$

Numerical results are displayed in Table III. In Fig. 3 the online prediction errors are shown. It can be seen that most of the prediction errors are varying between -0.05 and 0.05. It takes 0.351014s for SEFS to compute the result.

Observed from Table III that the numbers of fuzzy rules used by eT2RFNN and DeTS are slightly smaller than that used by SEFS. However, considering the complexity of the system structure and the accuracy achieved, SEFS still has its advantages. The specific analysis is shown below. i) Generalized Gaussian membership function is implemented in eT2RFNN, which makes eT2RFNN has a more complicated structure with more additional parameters get involved to measure the relationship between the data. In this aspect, SEFS enables more accurate predictions while using a much more simple system. ii) Although DeTS has a slightly more simple structure than SEFS, the proposed algorithm improves the prediction accuracy significantly from 0.440 to 0.1287. Although both DeTS and SEFS apply a small amount of fuzzy rules and simple structures, this significant improvement on the accuracy is worth with this slight increase of the complexity.

TABLE IV
EXAMPLE 3: ONLINE PREDICTION OF S&P 500 DAILY CLOSING PRICE.

	No. of rules (AVG.)	No. of parameters (AVG.)	NDEI
SEFS	2(1.2835)	32(20.5360)	0.0182
eTS [24]	14	75	0.04
Simpl_eTS [25]	7	39	0.04
PANFIS [22]	4	144	0.09
GENEFIS [23]	2	72	0.07
eT2RFNN [28]	2	110	0.04

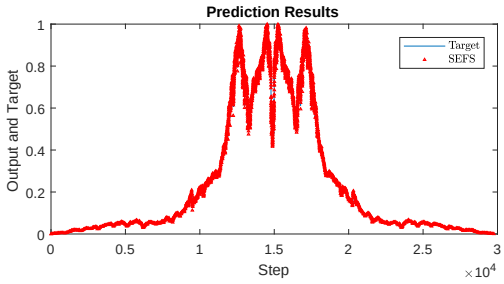


Fig. 4. Online prediction results for example 3.

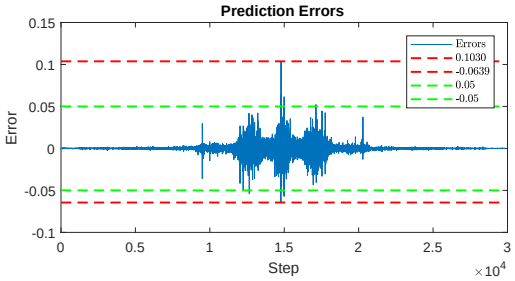


Fig. 5. Prediction errors for example 3.

D. Example 3: Online prediction of S&P 500 daily closing price

This data set contains 60 years daily closing price of S&P500 collected from Yahoo! Finance website ranging from 03.01.1950 to 12.03.2009. There are totally 14893 data points. We use SEFS to make online predictions for the original time series and the flipped time series with 29786 data points. The prediction model is of the form (38),

$$\hat{x}(t+1) = f(x(t-4), x(t-3), x(t-2), x(t-1), x(t)). \quad (38)$$

The original data set is used for training and the flipped time series is applied for testing. The results are listed in Table IV. Fig. 4 and Fig. 5 demonstrate SEFS predicts the time series preciously. Furthermore, the maximum absolute error is 0.1030 and most of the prediction errors are located between -0.05 and 0.05, and the computing time of SEFS is 2.346683s.

V. CONCLUSION

This paper proposes an online learning algorithm known as SEFS for solving regression problems of data stream. The major novelties of SEFS are its new fuzzy rule adding

and merging methodologies to evolve the structure of the fuzzy rule base. To be more specific, on one hand, an error based dynamic threshold, which can learn and update itself automatically, is proposed for adding new fuzzy rules. This not only enables the frequency of adding the fuzzy rules to be controlled by the dynamics of the data stream, but also avoid both over-fitting and under-fitting phenomenon. On the other hand, a geometric based similarity measure is developed in this paper. A direct, accurate and easily computed analytic formula of the L^2 distance between firing strengths of two rules is inducted and proposed to deal with the difficulty in calculating the L^2 distance, when membership functions are Gaussian type. This similarity measure leads to a more relax merging criterion but reserves the accuracy. Furthermore, consequent parameters are updated by an improved WRLS method, a local version of VFF-RLS — VFF-WRLS. The experimental results based on three benchmark examples show that SEFS outperforms many of the existing state-of-the-art algorithms in accuracy.

Our future work mainly focus on the following aspects. i) To further prevent over-fitting, *rule pruning method* will be considered by pruning fuzzy rules which have small number of observations in the corresponding clusters. ii) Generalized Gaussian membership functions will be implemented to further improve the prediction accuracy. iii) Completely self-learned and self-evolved control parameters will be investigated.

APPENDIX A
THE PROOF OF PROPOSITION III.1

Proof. Let $A = \sum_{k=1}^{t} \lambda^{t-k} e_k$, then $\varepsilon_t = \varepsilon_{max} - (\varepsilon_{max} - \varepsilon_{min})\exp\{-A\}$. Assume $A' \le A''$.

$$\begin{aligned}\varepsilon_t' - \varepsilon_t'' =& \varepsilon_{max} - (\varepsilon_{max} - \varepsilon_{min})\exp\{-A'\} \\ & - [\varepsilon_{max} - (\varepsilon_{max} - \varepsilon_{min})\exp\{-A''\}] \\ =& -(\varepsilon_{max} - \varepsilon_{min})\exp\{-A'\}(1 - \exp\{A' - A''\}) \quad (39)\end{aligned}$$

$\because A' - A'' \le 0, \therefore \exp\{A' - A''\} \le 1$, thus, $\varepsilon_t' - \varepsilon_t'' \le 0$. Therefore, ε_t is monotonically increasing against $\sum_{k=1}^{t} \lambda^{t-k} e_k$. □

APPENDIX B
INDUCTION OF (9) AND (10) IN *rule updating method*

Suppose $x(t)$ is the new input and the cluster center and radius for fuzzy rule R_{i^*} is $c_{i^*}(t-1) = (c_{i^*,1}(t-1), c_{i^*,2}(t-1), \ldots, c_{i^*,n}(t-1))$ and $\sigma_{i^*}(t-1) = (\sigma_{i^*,1}(t-1), \sigma_{i^*,2}(t-1), \ldots, \sigma_{i^*,n}(t-1))$, respectively. Assume the input points which satisfy *rule updating method* before $x(t)$ comes are $\{x'(k)\}_{k=1}^{N_{i^*}(t-1)}$. Input $x(t)$ satisfies the condition of *rule updating method*, then the updated center $(c_{i^*}(t))$ and radius $(\sigma_{i^*}(t))$ of rule R_{i^*} are inducted by (40) and (41), respectively.

$$\begin{aligned}c_{i^*}(t) =& \Big(\sum_{k=1}^{N_{i^*}(t-1)} x'(k) + x(t)\Big)/(N_{i^*}(t-1)+1) \\ =& (N_{i^*}(t-1)\cdot c_{i^*}(t-1) + x(t))/(N_{i^*}(t-1)+1) \\ =& c_{i^*}(t-1) + (x(t) - c_{i^*}(t-1))/(N_{i^*}(t-1)+1)\end{aligned} \quad (40)$$

$$\begin{aligned}(\sigma_{i^*,j}(t))^2 =& \frac{\sum_{k=1}^{N_{i^*}(t-1)}(x_j'(k) - c_{i^*,j}(t))^2 + (x_j(t) - c_{i^*,j}(t))^2}{N_{i^*}(t-1)+1} \\ =& \frac{\sum_{k=1}^{N_{i^*}(t-1)}(x_j'(k) - c_{i^*,j}(t-1) + c_{i^*,j}(t-1) - c_{i^*,j}(t))^2}{(N_{i^*}(t-1)+1)} \\ & + \frac{(x_j(t) - c_{i^*,j}(t))^2}{N_{i^*}(t-1)+1} \\ =& (\sigma_{i^*,j}(t-1))^2 + \frac{(x_j(t) - c_{i^*,j}(t))^2 - (\sigma_{i^*,j}(t-1))^2}{N_{i^*}(t-1)+1} \\ & + \frac{N_{i^*,j}(t-1)(c_{i^*,j}(t) - c_{i^*,j}(t-1))^2}{N_{i^*}(t-1)+1},\end{aligned} \quad (41)$$

where $j = 1, 2, \ldots, n$.

APPENDIX C
INDUCTION OF (21) AND (22) IN *rule merging method*

Assume that all the historical inputs within the cluster of rule $R_{i_{k^*}}$ are $\{x_{i_{k^*}}(l_{k^*})\}_{l_{k^*}=1}^{N_{i_{k^*}}}$. Then, c_{i^*}' and σ_{i^*}' could be computed by (42) and (43), respectively. The induction of ψ_{i^*}' is almost the same as c_{i^*}'.

$$c_{i^*}' = \frac{\sum_{l_{i^*}=1}^{N_{i^*}} x_{i^*}(l_{i^*}) + \sum_{l_{k^*}=1}^{N_{i_{k^*}}} x_{i_{k^*}}(l_{k^*})}{N_{i^*} + N_{i_{k^*}}} = \frac{N_{i^*} c_{i^*} + N_{i_{k^*}} c_{i_{k^*}}}{N_{i^*} + N_{i_{k^*}}}, \quad (42)$$

$$\begin{aligned}(\sigma_{i^*}')^2 =& \frac{\sum_{l_{i^*}=1}^{N_{i^*}}(x_{i^*}(l_{i^*}) - c_{i^*})^2 + \sum_{l_{k^*}=1}^{N_{i_{k^*}}}(x_{i_{k^*}}(l_{i_{k^*}}) - c_{i_{k^*}})^2}{N_{i^*} + N_{i_{k^*}}} \\ =& \{\sum_{l_{i^*}=1}^{N_{i^*}}(x_{i^*}(l_{i^*}) - c_{i^*}' + c_{i^*}' - c_{i^*})^2 + \sum_{l_{k^*}=1}^{N_{i_{k^*}}}(x_{i_{k^*}}(l_{i_{k^*}}) \\ & - c_{i^*}' + c_{i^*}' - c_{i_{k^*}})^2\}/\{N_{i^*} + N_{i_{k^*}}\} \\ =& \{N_{i^*}\sigma_{i^*}^2 + N_{i_{k^*}}\sigma_{i_{k^*}}^2 + N_{i^*}(c_{i^*}' - c_{i^*})^2 + N_{i_{k^*}}(c_{i^*}' - c_{i_{k^*}})^2\} \\ & /\{N_{i^*} + N_{i_{k^*}}\}.\end{aligned} \quad (43)$$

APPENDIX D
THE PROOF OF THEOREM III.1

Proof. Without loose of generality, we prove the theorem from the following three cases.

1) *Rule adding:* Assume $\Omega' \subset \Omega$ is a closed subset and $\Omega' \subset \cup_{i=1}^{n_0} B_i(c_i, k_i\sigma_i)$, where $B_i(c_i, k_i\sigma_i)$ are the open balls with centers c_i and radiuses $k_i\sigma_i$.
   (i) Assume there exist countable number of points $\{x(k)\}_{k=1}^{\infty} = \Omega \setminus \Omega'$. Based on the rule adding method, $\forall x(k)$, there would be a new fuzzy rule added with center $x(k)$ and radius $\frac{1}{k_t}\|x(k) - c_{i^*}\|$, where $k_t = \sqrt{-2\log\varepsilon_t}$, $i^* = \arg\max_{i=1,\ldots,n_0} \|x(k) - c_i\|$. Then, $x(k) \in B_k'(x(k), k_t\sigma_k)$, where $B_k'(x(k), k_t\sigma_k)$ is an open ball with center $x(k)$. Besides, as $\varepsilon_t \ge \varepsilon_{min}$, $\therefore \exp\{-\sum_{j=1}^{n} \frac{(x_j(k) - c_{k,j})^2}{2\sigma_{k,j}^2}\} \ge \varepsilon_{min}$. Therefore, $\{\cup_{i=1}^{n_0} B_i(c_i, k_i\sigma_i)\} \cup \{\cup_{k=1}^{\infty} B'(x(k), k_t\sigma_k)\}$ is an open cover of Ω. $\because \Omega$ is compact, $\therefore \exists\, N_0' \in \mathbb{N}, N_0' < \infty$ s.t.

$\Omega \subset \{\cup_{i=1}^{n_0} B_i(c_i, k_i\sigma_i)\} \cup \{\cup_{l=1}^{N_0'} B'(x(l), k_t\sigma_l)\}$. Let $N_0 = n_0 + N_0'$, the result is proved.

(ii) Assume there exists a closed subset $\Omega'' = \Omega \setminus \Omega'$. Construct a bounded countable subset $W = \mathbb{Q}^n \cap \Omega''$. It is obvious that W is dense in Ω''. Based on the rule adding method, $\forall\ x'(k) \in W$, $\exists\ B_k'(x'(k), k_t\sigma_k)$ with k_t, σ_k are the same as they are in (i). As W is countable, so $\{B_k'(x'(k), k_t\sigma_k)\}_{k=1}^{\infty}$ is a countable open cover of W. Because W is dense in Ω'', so $\forall x \in \Omega''$, $\exists x'(k) \in W$ s.t. $x \in B_k'(x'(k), k_t\sigma_k)$. Therefore, $\Omega'' \subset \cup_{k=1}^{\infty} B_k'(x'(k), k_t\sigma_k)$. Further, similar to (i), $\because\ \Omega''$ is compact, $\exists\ N_0' < \infty$ s.t. $\Omega \subset \{\cup_{i=1}^{n_0} B_i(c_i, k_i\sigma_i)\} \cup \{\cup_{l=1}^{N_0'} B_l'(x'(l), k_t\sigma_l)\}$.

2) *Rule updating:*
As $\lim_{N_{i^*}\to\infty} c_{i^*}(t) = c_{i^*}$, $\lim_{N_{i^*}\to\infty} \sigma_{i^*}^2(t) = \sigma_{i^*}^2$, and (9) and (10), then $\lim_{N_{i^*}\to\infty} c_{i^*}(t+1) = c_{i^*}$, $\lim_{N_{i^*}\to\infty} \sigma_{i^*}^2(t+1) = \sigma_{i^*}^2$. Based on the convergency of the cluster centers and radiuses, the conclusion could be get directly.

3) *Rule merging:*
As $\exists i_1, i_2 \in \{1, 2, \ldots, K\}$ s.t. $\forall \varepsilon > 0$, $\| \gamma_{i_1}(x) - \gamma_{i_2} \|_{L^2} < \varepsilon$, then, $\forall x \in \Omega$, $|\prod_{j=1}^{n} \mu_{i_1,j}(x) - \prod_{j=1}^{n} \mu_{i_2,j}(x)| = |\exp\{-\sum_{j=1}^{n} \frac{(x_j - c_{i_1,j})^2}{2\sigma_{i_1,j}^2}\} - \exp\{-\sum_{j=1}^{n} \frac{(x_j - c_{i_2,j})^2}{2\sigma_{i_2,j}^2}\}| < \varepsilon$ holds. So $|\sum_{j=1}^{n} \{(x_j - c_{i_1,j})^2\sigma_{i_2,j}^2 - (x_j - c_{i_2,j})^2\sigma_{i_1,j}^2\}| < \varepsilon$, besides, $x_j = c_{i_1,j} \Longrightarrow \|c_{i_1} - c_{i_2}\| < \varepsilon$, $x_j = c_{i_1,j} + \sigma_{i_1,j} \Longrightarrow \|\sigma_{i_2}^2 - \sigma_{i_1}^2\| < \varepsilon$. Further, $\because \lim_{N_{i_1}\to\infty} c_{i_1}(t) = c_{i_0}$, $\lim_{N_{i_2}\to\infty} c_{i_2}(t) = c_{i_0}$, $\lim_{N_{i_1}\to\infty} \sigma_{i_1}^2(t) = \sigma_{i_0}^2$, $\lim_{N_{i_2}\to\infty} \sigma_{i_2}^2(t) = \sigma_{i_0}^2$, (21), (22), $\therefore \lim_{N_{i_1}, N_{i_2}\to\infty} c_{i_0}(t+1) = c_{i_0}$, $\lim_{N_{i_1}, N_{i_2}\to\infty} \sigma_{i_0,j}^2(t+1) = \sigma_{i_0,j}^2$. Then, the conclusion is obtained.

□

APPENDIX E
INDUCTION OF VFF-WRLS

For each rule R_i, suppose

$$\bar{R}_i(t) = \sum_{k=1}^{t} \lambda_i^{t-k} xe(k) xe^T(k) \quad \text{and} \tag{44}$$

$$Q_i(t) = \sum_{k=1}^{t} \lambda_i^{t-k} \theta_i(k) y(k) xe(k). \tag{45}$$

Then, similar to [42], the conventional recursive least square (CRLS) updating formulas for ψ_i could be presented by (46) and (47),

$$P_i(t) = \frac{1}{\lambda_i}\left(P_i(t-1) - \frac{\theta_i(t) P_i(t-1) xe(t) xe^T(t) P_i(t-1)}{\lambda_i + \tau_i(t)}\right), \tag{46}$$

$$\psi_i(t) = \psi_i(t-1) + \frac{q\theta_i(t) P_i(t-1) xe(t) xe^T(t) P_i(t-1)}{\lambda_i + \tau_i(t)}, \tag{47}$$

where $\tau_i(t) = \theta_i(t) xe^T(t) P_i(t-1) xe(t)$, q is the convergence factor. So the posterior error signal $\varepsilon_i(t)$ can be expressed by (48),

$$\begin{aligned}\varepsilon_i(t) =& \theta_i(t)[y(t) - xe^T(t)\psi_i(t)] \\ =& (1 - q\tau_i'(t)) e_i(t) - q\tau_i'(t)\nu_i(t),\end{aligned} \tag{48}$$

where $\tau_i'(t) = \frac{\tau_i(t)}{\lambda_i + \tau_i(t)}$, $e_i(t) = e_i'(t) + \nu_i(t) = \theta_i(t)[xe^T(t)h(t-1) - xe^T(t)\psi_i(t-1)]$. The optimal λ_i are obtained through the same way as [42] by minimizing $E[(\varepsilon_i'(t))^2]$ and setting $q = 1$.

ACKNOWLEDGMENT

The authors would like to thank the reviewers and the Associate Editor for their constructive and very helpful comments.

REFERENCES

[1] L. Pietruczuk, L. Rutkowski, M. Jaworski, and P. Duda, "How to adjust an ensemble size in stream data mining?" *Information Sciences*, vol. 381, pp. 46–54, 2017.

[2] E. Lughofer and M. Pratama, "On-line active learning in data stream regression using uncertainty sampling based on evolving generalized fuzzy models," *IEEE Transactions on Fuzzy Systems*, 2017.

[3] N. K. Kasabov and Q. Song, "Denfis: dynamic evolving neural-fuzzy inference system and its application for time-series prediction," *IEEE transactions on Fuzzy Systems*, vol. 10, no. 2, pp. 144–154, 2002.

[4] E. D. Lughofer, "Flexfis: A robust incremental learning approach for evolving takagi–sugeno fuzzy models," *IEEE Transactions on fuzzy systems*, vol. 16, no. 6, pp. 1393–1410, 2008.

[5] R. D. Baruah and P. Angelov, "Dec: Dynamically evolving clustering and its application to structure identification of evolving fuzzy models," *IEEE transactions on cybernetics*, vol. 44, no. 9, pp. 1619–1631, 2014.

[6] A. Lemos, W. Caminhas, and F. Gomide, "Multivariable gaussian evolving fuzzy modeling system," *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 1, pp. 91–104, 2011.

[7] E. Lughofer, C. Cernuda, S. Kindermann, and M. Pratama, "Generalized smart evolving fuzzy systems," *Evolving Systems*, vol. 6, no. 4, pp. 269–292, 2015.

[8] L. Maciel, R. Ballini, and F. Gomide, "An evolving possibilistic fuzzy modeling approach for value-at-risk estimation," *Applied Soft Computing*, 2017.

[9] G. Leng, T. M. McGinnity, and G. Prasad, "An approach for on-line extraction of fuzzy rules using a self-organising fuzzy neural network," *Fuzzy sets and systems*, vol. 150, no. 2, pp. 211–243, 2005.

[10] J. de Jesús Rubio, "Sofmls: online self-organizing fuzzy modified least-squares network," *IEEE Transactions on Fuzzy Systems*, vol. 17, no. 6, pp. 1296–1309, 2009.

[11] E. Lughofer, M. Pratama, and I. Skrjanc, "Incremental rule splitting in generalized evolving fuzzy systems for autonomous drift compensation," *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 4, pp. 1854–1865, 2018.

[12] J. de Jesús Rubio, "Usnfis: uniform stable neuro fuzzy inference system," *Neurocomputing*, vol. 262, pp. 57–66, 2017.

[13] W. L. Tung and C. Quek, "efsm-a novel online neural-fuzzy semantic memory model," *IEEE Transactions on Neural Networks*, vol. 21, no. 1, pp. 136–157, 2010.

[14] A. M. Silva, W. Caminhas, A. Lemos, and F. Gomide, "A fast learning algorithm for evolving neo-fuzzy neuron," *Applied Soft Computing*, vol. 14, pp. 194–209, 2014.

[15] N. N. Nguyen, W. J. Zhou, and C. Quek, "Gsetsk: a generic self-evolving tsk fuzzy neural network with a novel hebbian-based rule reduction approach," *Applied Soft Computing*, vol. 35, pp. 29–42, 2015.

[16] D. Ge and X.-J. Zeng, "Learning evolving ts fuzzy systems with both local and global accuracy–a local online optimization approach," *Applied Soft Computing*, 2017.

[17] G.-B. Huang, P. Saratchandran, and N. Sundararajan, "A generalized growing and pruning rbf (ggap-rbf) neural network for function approximation," *IEEE Transactions on Neural Networks*, vol. 16, no. 1, pp. 57–67, 2005.

[18] H. Rong, N. Sundararajan, G. Huang, and G. Zhao, "Extended sequential adaptive fuzzy inference system for classification problems," *Evolving Systems*, vol. 2, no. 2, pp. 71–82, 2011.

[19] J. de Jesús Rubio and A. Bouchachia, "Msafis: an evolving fuzzy inference system," *Soft Computing*, vol. 21, no. 9, pp. 2357–2366, 2017.

[20] J. de Jesús Rubio, "Error convergence analysis of the sufin and csufin," *Applied Soft Computing*, 2018. [Online]. Available: http://dx.doi.org/10.1016/j.asoc.2018.04.003

[21] H.-J. Rong, N. Sundararajan, G.-B. Huang, and P. Saratchandran, "Sequential adaptive fuzzy inference system (safis) for nonlinear system identification and prediction," *Fuzzy sets and systems*, vol. 157, no. 9, pp. 1260–1275, 2006.

[22] M. Pratama, S. G. Anavatti, P. P. Angelov, and E. Lughofer, "Panfis: a novel incremental learning machine," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 1, pp. 55–68, 2014.

[23] M. Pratama, S. G. Anavatti, and E. Lughofer, "Genefis: toward an effective localist network," *IEEE Transactions on Fuzzy Systems*, vol. 22, no. 3, pp. 547–562, 2014.

[24] P. P. Angelov and D. P. 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.

[25] P. Angelov and D. Filev, "Simpl_ets: A simplified method for learning evolving takagi-sugeno fuzzy models," in *Fuzzy Systems, 2005. FUZZ'05. The 14th IEEE International Conference on*. IEEE, 2005, pp. 1068–1073.

[26] P. Angelov, "Evolving takagi-sugeno fuzzy systems from streaming data (ets+)," *Evolving intelligent systems: methodology and applications*, vol. 12, p. 21, 2010.

[27] L. Maciel, F. Gomide, and R. Ballini, "Enhanced evolving participatory learning fuzzy modeling: an application for asset returns volatility forecasting," *Evolving Systems*, vol. 2, no. 5, pp. 75–88, 2014.

[28] M. Pratama, J. Lu, E. Lughofer, G. Zhang, and M. J. Er, "Incremental learning of concept drift using evolving type-2 recurrent fuzzy neural network," *IEEE Transactions on Fuzzy Systems*, 2016.

[29] P. Angelov, "Fuzzily connected multimodel systems evolving autonomously from data streams," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 41, no. 4, pp. 898–910, 2011.

[30] R. J. Oentaryo, M. J. Er, S. Linn, and X. Li, "Online probabilistic learning for fuzzy inference system," *Expert Systems with Applications*, vol. 41, no. 11, pp. 5082–5096, 2014.

[31] M. Setnes, R. Babuska, U. Kaymak, and H. R. van Nauta Lemke, "Similarity measures in fuzzy rule base simplification," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 28, no. 3, pp. 376–386, 1998.

[32] S. W. Tung, C. Quek, and C. Guan, "et2fis: An evolving type-2 neural fuzzy inference system," *Information Sciences*, vol. 220, pp. 124–148, 2013.

[33] C. Lin and C. G. Lee, "Real-time supervised structure/parameter learning for fuzzy neural network," in *IEEE Conf. on Fuzzy Systems*, 1992, pp. 1283–1291.

[34] M.-Y. Chen and D. Linkens, "Rule-base self-generation and simplification for data-driven fuzzy models," *Fuzzy Sets and Systems*, vol. 142, no. 2, pp. 243–265, 2004.

[35] M. Pratama, M. Er, X. Li, R. Oentaryo, E. Lughofer, and I. Arifin, "Data driven modeling based on dynamic parsimonious fuzzy neural network," *Neurocomputing*, vol. 110, pp. 18–28, 2013.

[36] S. Wu, M. J. Er, and Y. Gao, "A fast approach for automatic generation of fuzzy rules by generalized dynamic fuzzy neural networks," *IEEE Transactions on Fuzzy Systems*, vol. 9, no. 4, pp. 578–594, 2001.

[37] C.-T. Chao, Y.-J. Chen, and C.-C. Teng, "Simplification of fuzzy-neural systems using similarity analysis," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 26, no. 2, pp. 344–354, 1996.

[38] J. Qiao, W. Li, X. Zeng, and H. Han, "Identification of fuzzy neural networks by forward recursive input-output clustering and accurate similarity analysis," *Applied Soft Computing*, vol. 49, pp. 524–543, 2016.

[39] M. Abramowitz and I. Stegun, *Handbook of mathematical functions: with formulas, graphs, and mathematical tables*. Courier Corporation, 1964, vol. 55.

[40] W. J. Cody, "Rational chebyshev approximations for the error function," *Mathematics of Computation*, vol. 23, no. 107, pp. 631–637, 1969.

[41] E. Lughofer, J. Bouchot, and A. Shaker, "On-line elimination of local redundancies in evolving fuzzy systems," *Evolving Systems*, vol. 2, no. 3, pp. 165–187, 2011.

[42] M. Z. A. Bhotto and A. Antoniou, "New improved recursive least-squares adaptive-filtering algorithms," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, no. 6, pp. 1548–1558, 2013.

[43] M. Zibulevsky and M. Elad, "L1-l2 optimization in signal and image processing," *IEEE Signal Processing Magazine*, vol. 27, no. 3, pp. 76–88, 2010.

[44] I. Daubechies, M. Defrise, and C. D. Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Communications on pure and applied mathematics*, vol. 57, no. 11, pp. 1413–1457, 2004.

[45] D. Coyle, G. Prasad, and T. McGinnity, "Faster self-organizing fuzzy neural network training and a hyperparameter analysis for a brain–computer interface," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 6, pp. 1458–1471, 2009.

[46] C. Juang and C. Hsieh, "A locally recurrent fuzzy neural network with support vector regression for dynamic-system modeling," *IEEE Transactions on Fuzzy Systems*, vol. 18, no. 2, pp. 261–273, 2010.

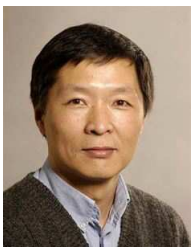
[47] M. Pratama, J. Lu, and G. Zhang, "Evolving type-2 fuzzy classifier," *IEEE Transactions Fuzzy Systems*, vol. 24, no. 3, pp. 574–589, 2016.

[48] R. Bao, H. Rong, P. P. Angelov, B. Chen, and P. k. Wong, "Correntropy-based evolving fuzzy neural system," *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 3, pp. 1324–1338, 2018.

Dongjiao Ge received the B.Sc. degree in mathematics and applied mathematics in 2012, and the M.Sc. degree in applied mathematics in 2015 from Sichuan University, Chengdu, China. She is currently pursuing her Ph.D. degree in computer science with the School of Computer Science, University of Manchester, Manchester, U.K.

Her current research interests include computational intelligence, machine learning, and statistical learning.

Miss Ge was awarded the President's Doctoral Scholar Award (Sept.2015 – Sept. 2018), which is a flagship funding scheme of the University of Manchester.



Xiao-Jun Zeng received the B.Sc. degree in Mathematics and the M.Sc. degree in Control Theory and Operation Research from Xiamen University, Xiamen, China, respectively and the Ph.D. degree in Computation from the University of Manchester, Manchester, U.K.

Dr. Zeng has been with the School of Computer Science at the University of Manchester since 2002, where he is currently a Senior Lecturer in Machine Learning and Optimisation. Before joining the University of Manchester in 2002, he was with Knowledge Support Systems, Ltd., Manchester, between 1996 – 2002, where he was the Head of Research, developing intelligent pricing decision support systems which won the European Information Society Technologies Award in 1999 and Microsoft European Retail Application Developer (RAD) Awards in 2001 and 2003. His research in intelligent pricing decision support systems was selected by UKCRC, CPHC, and BCS Academy as one of 20 impact cases to highlight the impact made by UK academic Computer Science Research within the UK and worldwide over the period 2008 – 2013.

Dr. Zeng's main research interests include computational intelligence, machine learning, big data, decision support systems, computational finance, energy demand management, and game theory.

Dr. Zeng has served to scholarly and professional communities in various roles including an Associate Editor of the IEEE Transactions on Fuzzy Systems between 2004 – 2018.