Evolving Systems
https://doi.org/10.1007/s12530-020-09334-5

ORIGINAL PAPER

An overview on evolving systems and learning from stream data

Daniel Leite[1] · Igor Škrjanc[2] · Fernando Gomide[3]

Received: 8 August 2019 / Accepted: 3 March 2020

Abstract
Evolving systems unfolds from the interaction and cooperation between systems with adaptive structures, and recursive methods of machine learning. They construct models and derive decision patterns from stream data produced by dynamically changing environments. Different components that assemble the system structure can be chosen, being rules, trees, neurons, and nodes of graphs amongst the most prominent. Evolving systems relate mainly with time-varying environments, and processing of nonstationary data using computationally efficient recursive algorithms. They are particularly appropriate for online, real-time applications, and dynamically changing situations or operating conditions. This paper gives an overview of evolving systems with focus on system components, learning algorithms, and application examples. The purpose is to introduce the main ideas and some state-of-the-art methods of the area as well as to guide the reader to the essential literature, main methodological frameworks, and their foundations.

Keywords Evolving intelligence · Fuzzy systems · Neural networks · Incremental machine learning · Online data stream · Adaptive systems

1 Introduction

The operation of evolving systems results from online sequential data processing to learn the nature of local subsystems and their interactions to endure self-organization of the system structure and parameters. The system should develop and update itself to unknown environments, and detect potential temporal shifts and drifts in input data. Applications are numerous especially in modeling and identification, control, prediction, clustering and classification, fault diagnosis, anomaly detection, frequent pattern mining, and recognition.

✉ Daniel Leite
daniel.leite@ufla.br

Igor Škrjanc
igor.skrjanc@fe.uni-lj.si

Fernando Gomide
gomide@dca.fee.unicamp.br

1 Department of Automatics, Federal University of Lavras, Lavras, MG 37200-000, Brazil

2 Faculty of Eletric Engineering, University of Ljubljana, Ljubljana, Slovenia

3 School of Electrical and Computer Engineering, University of Campinas, Campinas, Brazil

Processing and modeling nonstationary stream data bring unique issues and challenges in online machine intelligence. For example, machines in industry and mobile robots suffer from stress, aging, and faults; economic indicators, such as stock indices, vary at a high speed; communication systems transmission capacity and responsiveness are subject to continuous changes; users behavior in a social network change over time. Computational models should be supplied with incremental learning algorithms to be able to evolve and deal with such changes. Evolution provides a system with flexibility to improve its short-term performance, and increases its chance to survive in the long-term despite of changes in the environment and in its components. While small changes in system parameters can be handled as a form of uncertainty, and be properly addressed by using parameter estimation mechanisms, changes in the system structure requires a higher level of adaptation and autonomy.

Prior studies on evolving systems were mainly concerned with neural networks (Fritzke 1994; Williamson 1996), fuzzy rule-based systems (Angelov and Filev 2004; Lughofer 2008) (evolving fuzzy systems), and neural-fuzzy hybrids (Kasabov and Song 2002). The structure of rule-based systems is identified by the nature and number of rules. For instance, evolving fuzzy rule-based systems may use linguistic fuzzy rules, functional fuzzy rules, or their combination (Leite 2012). The structure of neuro-fuzzy systems is, in turn, recognized by the nature of the neurons and their connections. Important milestones in the history of evolving systems include the publication of the monographs Evolving Connectionist Systems (Kasabov 2007), Evolving Intelligent Systems (Angelov et al. 2010), and Evolving Fuzzy Systems: Methodologies, Advanced Concepts and Applications (Lughofer 2011). The journal entitled Evolving Systems by Springer started in 2010. A recent survey article on evolving fuzzy and neuro-fuzzy methods for clustering, regression, identification, and classification is found in Škrjanc et al. (2019).

Recently, advances and generalizations of the pioneering studies are found in the realm of statistical models (Agrawal and Bala 2008; Hisada et al. 2010; Škrjanc 2009; Dovžan and Škrjanc 2011) granular computing (Leite et al. 2012, 2013, 2015, 2019), deep learning (Pratama and Wang 2019; Prasad et al. 2018), regression trees (Hapfelmeier et al. 2014; Lemos et al. 2011a), support vector machines (Bordes and Bottou 2005), type-2 fuzzy systems (Tung et al. 2013; Za'in et al. 2017; Pratama et al. 2016), interval mathematics (Leite et al. 2012, 2010), spiking neural networks (Doborjeh et al. 2018), and ensemble learning (Leite and Škrjanc 2019; Heeswijk et al. 2009; Lughofer and Buchtala 2013; Iglesias et al. 2014; Pratama et al. 2018; Soares et al. 2018). Figure 1 shows the essential categories of evolving approaches that have been developed. Applications in intelligent sensors and actuators (Angelov and Kordon 2010; Angelov et al. 2008), autonomous unmanned systems (Yourdshahi et al. 2018; Angelov et al. 2008; Klančar and Škrjanc 2015), industrial process monitoring (Filev and Tseng 2006; Wang and Vrbanek 2008; Lughofer 2008; Lemos et al. 2010), biomedical data processing (Kasabov 2007; Leite et al. 2013), (Škrjanc 2015), cyber security (Škrjanc et al. 2018), real-time financial analysis (Maciel et al. 2018, 2017), weather forecasting (Soares et al. 2018; Leite et al. 2012), smart grids (Silva et al. 2018, 2018), web news mining (Za'in et al. 2017), tracking of chaotic systems (Leite et al. 2016) and the Katrina, Sandy and Wilma cyclones (Soares et al. 2018), analysis of electro-oculography and encephalography signals (Rubio and Bouchachia 2017; Doborjeh et al. 2018; Rubio 2014), seismocardiogram-based monitoring during physical activities (Malcangi et al. 2018), online identification of quadcopter hovering dynamics (Ferdaus et al. 2019), missing data imputation (Garcia et al. 2019), finger dynamics modeling (Precup et al. 2018), recognition of drivers' actions (Škrjanc et al. 2018); in general control problems (Andonovski et al. 2016; Zdešar et al. 2014; Blažič et al. 2014), and in many other areas have been reported. Model-based evolving control design and closed-loop Lyapunov stability are achieved in Leite et al. (2015). Evolving audiovisual speech recognition is reported in Malcangi and Grew (2017).

Nonstationary data-stream processing poses questions that are not easily answered by many of the current commonly-used computational intelligence and machine learning methods because the latter require an offline batch-learning stage. This is because data streams are characterized by the following aspects: (i) samples in a stream arrive continuously; (ii) the system has no control over the order in which data samples arrive; (iii) streams are typically unbounded, and samples are sequentially recorded as long as the system operates; and (iv) a sample should ideally be discarded after being processed to avoid scalability issues. Ideally, computational models should be promptly updated to changing situations occurring over the life-time of a stream.

The effect of concept drift and shift in models and learning algorithms may be enormous. While the impact of concept drift can be suppressed using, e.g., model parameter adaptation procedures, concept shift may require searching in a hypothesis space. The key difference of evolving systems to online incremental machine learning is the ability of the former to simultaneously manage different kinds of

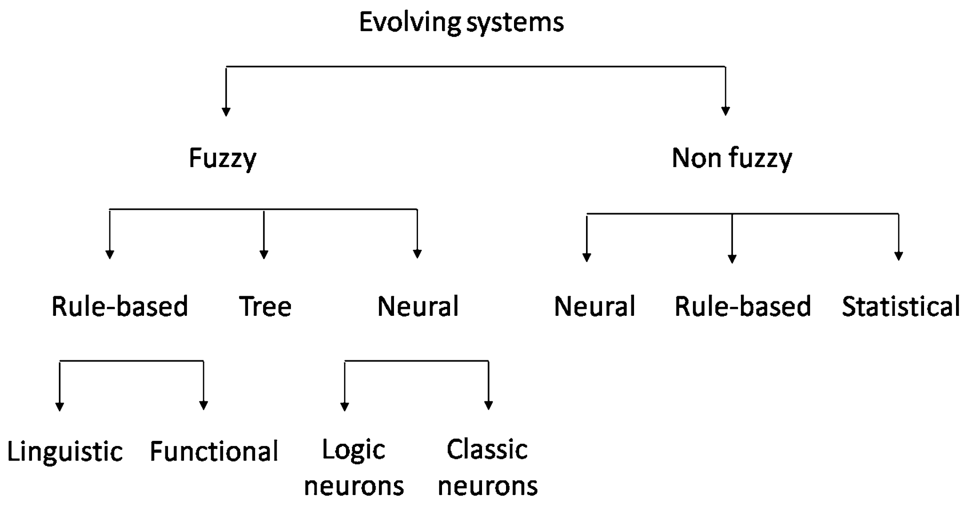


Fig. 1 Categories of evolving systems

changes (drift, shift, non-stationary behavior, environmental changes, etc.) by relying on parameter and structural updating procedures that scan a data sample only once (single-pass incremental learning). Contrariwise, typically, only parameters are updated in 'incremental machine learning'; that is, no intrinsic structural change of the model is carried out. Therefore, the term "evolving" means systems that are able to simultaneously learn and adapt their structure and parameters endlessly (Angelov and Zhou 2006). We also contrast the term "evolving" with the term "evolutionary", as used in genetic algorithms and genetic programming. Evolutionary processes proceed with populations of individuals using recombination, mutation, and selection mechanisms during generations (usually in a static and offline optimization context). "Adaptive" systems in the control and dynamical systems theory deal predominantly with parameter estimation, and, therefore, are also different than evolving systems. Evolving systems and models advance autonomously over time during the life span of the system.

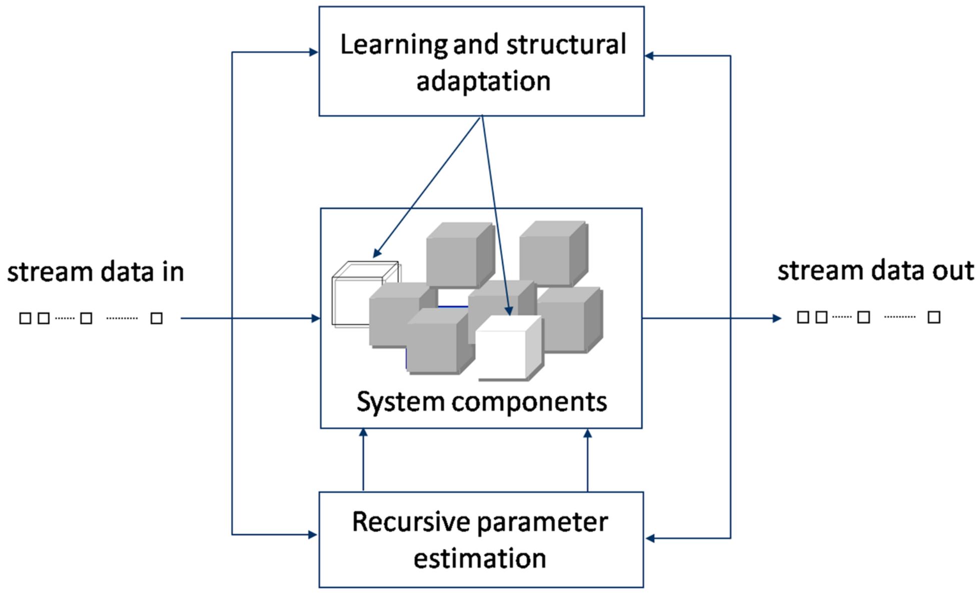
2 Fuzzy rule-based and neuro-fuzzy evolving systems

The issue of adapting models' structure and parameters automatically dates from the early 90's, especially for neural networks (Fritzke 1994; Williamson 1996; Kwok and Yeung 1997). Rule-based evolving systems have advantages over evolving black-box models, such as neural networks, because they are more transparent and linguistically explainable (Angelov 2010). The demand of knowledge from data often entails interpretability of models. Rule-based models—a class of models in which evolving fuzzy, interval, and granular-computing systems belong to—typically offer better interpretable insights in a given application domain compared to pure neural network or deep learning modeling approaches.

There are several studies in the literature proposing evolving fuzzy models capable of addressing problems like system identification, time series forecasting, pattern classification, process control, and so on. Many of these studies propose functional fuzzy models, e.g., (Angelov 2002; Angelov and Filev 2004; Lughofer 2008; Lima et al. 2010), which are based on a set of algebraic Takagi-Sugeno rules (Takagi and Sugeno 1985). Some studies propose combining Takagi-Sugeno and Mamdani consequent terms, e.g., (Leite et al. 2012, 2019, 2013), to give numerical estimations and an enclosure around them, which is sometimes called "granular" estimation, and may come associated with a linguistic value that adds interpretability to the model. Evolving fuzzy rule-based models employ unsupervised recursive clustering algorithms to update the antecedent part of some of their rules whenever new data arise. However, creating a new rule is always a possibility in case a data sample conveys significantly different information. Figure 2 shows the idea of structural and parametric adaptation in evolving systems in general. In particular, evolving fuzzy rule-based systems are evolving systems in which system components are fuzzy rules. For functional fuzzy rules, consequent parameters are found using recursive least-squares algorithms or variations (Ljung 1999).

One of the firsts evolving modeling framework is called evolving Takagi-Sugeno (Angelov and Filev 2004). This is a functional fuzzy rule-based model whose structure is continuously adapted using recursive clustering, and recursive least squares. The idea is to assign to each cluster a local fuzzy

Fig. 2 Evolving systems


Learning and structural adaptation
stream data in
stream data out
System components
Recursive parameter estimation

functional model. The collection of fuzzy rules, and their corresponding parameters, assemble the overall model. Many similar approaches can be found in the literature, e.g., (Lughofer 2008; Lima et al. 2010; Klančar and Škrjanc 2015). They differ mostly in the way that recursive clustering is performed. An example is the use of the participatory learning paradigm for recursive clustering (Lima et al. 2010). A multivariable evolving approach using the participatory learning approach was developed in Lemos et al. (2011b) to account for interactions among attributes, and to attenuate the curse of dimensionality during clustering. In Škrjanc and Dovžan (2015), the evolution of possibilistic Gustafson-Kessel hyperellipsoids were proposed with the purpose of capturing the covariance among attributes in an incremental basis. In Leite et al. (2019), a multi-objective function is used to guide the placement of granules in the data space, and simultaneously to maximize the specificity and coverage of local models. Evolving fuzzy regression trees are given in Lemos et al. (2011a). A tree uses an affine function on each leaf, whose parameters are adapted using the conventional least-squares algorithm. The tree structure is updated using a statistical selection procedure based on a hypothesis test. Improvements on the tree structure by using online incremental pruning methods are rare. Guarded Incremental Pruning (GuIP) is proposed in Hapfelmeier et al. (2014) to improve tree-based models in the sense of overfitting and overly large structures avoidance. Sub-trees that do not provide a significant contribution to the estimations are detected and removed.

Many other evolving methods proposed over the last 16 years can be mentioned. Some important methods based on rule-based models are: evolving Takagi-Sugeno (eTS) (Angelov and Filev 2004; Precup et al. 2018), extended Takagi-Sugeno (xTS and +eTS) (Angelov and Zhou 2006; Angelov 2010), FLEXible Fuzzy Inference System (FLEXFIS and FLEXFIS+) (Lughofer 2008; Lughofer et al. 2011), Generalized Smart Evolving Fuzzy System (GS-EFS) (Lughofer et al. 2015), Interval-Based evolving Modeling (IBeM) (Leite et al. 2012, 2010), Fuzzy-set-Based evolving Modeling (FBeM) (Leite et al. 2012), evolving Fuzzy Model (eFuMo) (Dovžan et al. 2015), Sequential Adaptive Fuzzy Inference System (SAFIS) (Rong et al. 2011), Modified Sequential Adaptive Fuzzy Inference System (MSAFIS) (Rubio and Bouchachia 2017), and evoling Optimal Granular System (eOGS) (Leite et al. 2019). All of these methods can be employed to supervised and semi-supervised classification or prediction under simple considerations. Evolving Classifiers (eClass, eClass0, and eClass1) are given in (Angelov and Zhou 2008), and the AutoClassify model in Angelov (2012). Typicality and Eccentricity-based Data Analytics (TEDA) is discussed in Kangin et al. (2015).

Evolving neural systems are systems for which the components shown in Figure 2 are neurons. Amongst neuro-fuzzy-based systems are the following: Evolving Fuzzy Neural Network (EFuNN) (Kasabov 2007), Dynamic Evolving Neuro-Fuzzy Inference System (DENFIS) (Kasabov and Song 2002), Self-Organizing Fuzzy Neural Networks (SOFNN) (Leng et al. 2004), Generalized Adaptive Neuro-Fuzzy Inference Systems (GANFIS) (Azeem et al. 2003), evolving Granular Neural Network (eGNN) (Leite et al. 2013; Leite 2019), Dynamic Fuzzy Neural Network (D-FNN) (Wu and Er 2000), Generalized Dynamic Fuzzy Neural Network (GD-FNN) (Wu et al. 2001), Neural Cube (NeuCube) (Kasabov 2014), online neuro-fuzzy ART-based model (NeuroFAST) (Tzafestas and Zikidis 2001), Evolving Self-Organizing Map (ESOM) (Kasabov 2007), Neural Gas (Fritzke 1994), Self-Organizing Fuzzy Modified Least Square network (SOFMLS) (Rubio 2009, 2017), Gath-Geva Evolving Neuro-Fuzzy Modeling (ENFM) (Soleimani-B et al. 2010), PArsimonious Network based on Fuzzy Inference System (PANFIS) (Pratama et al. 2014), and Recurrent Interval-Valued Metacognitive Scaffolding Fuzzy Neural Network (RIVMcS-FNN) (Pratama et al. 2015). The structure of neuro-fuzzy systems is recognized by the nature of the neurons, the network topology, and the number of neurons in the hidden layers.

The majority of the evolving neuro-fuzzy methods for regression is based on local radial-basis-function models (RBF models) or generalized constructions. The basic RBF models use Gaussian membership functions with equal spread, as in Wu et al. (2001). Other models employ ellipsoidal functions and, therefore, allow membership functions to have different widths in different dimensions. Ellipsoidal approaches are given, for example, in GD-FNN (Wu et al. 2001), and in SOFNN (Leng et al. 2004). In eGNN (Leite et al. 2013), hyper-rectangles and trapezoidal membership functions with different widths are used. Common to all frameworks is that neurons, connections, and local models are created, updated, merged, and deleted based on information uncovered from numerical or uncertain data streams. Fuzzy rules can be extracted from neuro-fuzzy models at any time. Distinctive classes of evolving neural networks with fuzzy neurons and fuzzy procedures for data processing include the neo-fuzzy neuron-based architectures of Silva et al. (2013) and Silva et al. (2015), and uninorm-based networks of Bordignon and Gomide (2014) and Leite et al. (2013). Recent developments on neo-fuzzy neurons and models are given in Bodyanskiy et al. (2016) and Bodyanskiy et al. (2018). A diversity of aggregation functions, such as T-norms, S-norms, averaging and compensatory operators, can be used in the body of a fuzzy neuron (Leite et al. 2013).

In general, the same evolving neuro-fuzzy methods originally proposed for regression, prediction, or control (which basically depends on the meaning and purpose of the output variable), can also be used in pattern recognition and classification by eliminating the layer of the neural network that represents locally-valid functions, and ignoring the respective adaptation equations. Otherwise, a classification problem can be handled as a regression problem by simply rounding the estimated output value to the nearest integer. Examples of evolving neuro-fuzzy networks used for classification include EFuNN, DENFIS, eGNN, and evolving TS neuro-fuzzy classifiers (Angelov 2010).

Ensemble learning has been discussed in evolving framework. In ensemble learning, multiple base models or experts are trained to solve the same problem (Polikar 2006). The diversity of the base models can improve overall estimation accuracy and model robustness by combining individual contributions. An ensemble of evolving eClass classifiers based on the concept of stacking is described in Iglesias et al. (2014). A fast deep learning network for handwriting recognition is proposed in Angelov and Gu (2017). The approach provides interpretable models. The network comprises an ensemble of zero-order evolving fuzzy rule-based models, which are built in parallel through an Autonomous Learning Multiple Model (ALMMo) method. Parsimonious ensemble (pENsemble) is proposed in Pratama et al. (2018) as an evolving variation of the dynamic weighted majority ensemble method by Kolter and Maloof (2007). An adaptive ensemble based on Extreme Learning Machines (ELMs) for one-step prediction is presented in Heeswijk et al. (2009). All-pairs evolving fuzzy models to handle online multiclass classification problems are proposed in Lughofer and Buchtala (2013). The approach can be viewed as an ensemble strategy that employs weighted voting based on a preference relation matrix to decide about the class of a sample. Three studies on ensembles of evolving predictors can be found in the literature (Bueno et al. 2015; Soares et al. 2018; Leite and Škrjanc 2019). In Bueno et al. (2015), an ensemble of Fuzzy-set-Based evolving Models (FBeM) (Leite et al. 2012) is outlined. In (Leite and Škrjanc 2019), ordered weighted averaging (OWA) aggregation functions is used to merge the contribution of individual optimal evolving granular experts (eOGS) (Leite et al. 2019). In Soares et al. (2018), an ensemble of cloud and evolving fuzzy models were combined through the weighted arithmetic mean to give weather estimations.

Common to all frameworks is that evolution should change the structure of the model that describes the behavior of the data stream, and update parameters associated to local models. As mentioned, the latter is generally handled by using some version of Weighted Recursive Least-Squares algorithm (Ljung 1999). The most challenging task, and also the basic ingredient of an evolving system, is therefore related to adding, deleting, splitting, and merging of clusters, neurons, granules, leaves, or clouds (local models in general), in order to assure significant flexibility in case of changing situations, and representability of the essence and dynamic of the data.

Regarding the creation of a new local model, usually, learning starts from scratch. Local models are added to the global model on the fly in order to expand the knowledge inherent to the model to new regions of interest in the data space. If a sample fulfills the conditions for the addition of a local model, then it usually defines the center of the local model. A second parameter to be defined is the size of the local model, which depends on the geometry of the representative object, which is given by a distance measure.

Merging is necessary if local models significantly overlap with each other as a result of updating mechanisms. The act of merging is sometimes called information fusion, and is usually caused by consecutive samples belonging to the gap between two or more local models, which are used to be disjoint previously. Merging is justified to eliminate redundancy. Splitting local models is defined for a finer structuring of the data space and model structure. Basically, an evolving algorithm should, in the case of regression and identification problems, accept a larger number of local models in the region of the data space in which the model output error (approximation or prediction error) is greater than the expected, or increases unexpectedly in a time interval.

Procedures to remove local models are convenient to be rid of elements that are no longer contributing to the model overall performance and understandability. Such procedures are of utmost importance in classification and pattern recognition to assure faster computation speed during data processing, and more compact rule bases and network topologies. In general, it happens that a local model is created in a part of the space where there are just a few representative samples. This may be justified by measurement errors or due to a change of the system behavior so that the local model is no longer useful after a number of iterations. These local models can be removed because they do not help in the description of the data. Nonetheless, care should be taken with seasonal behaviors since a local object may be reactivated in a further iteration. Moreover, in anomaly detection problems, unusual and idle representative objects may be more important than those highly operative local models, and therefore should not be removed.

Although creation, merging, splitting, and removing local models assure some kind of homogeneity and compactness of the overall structure, there are several issues to be addressed by the community to assure: (i) more elegant model architectures for high-performance computing; (ii) a more precise control over the flexibility and geometry of representative objects; (iii) higher robustness to outliers, missing data, and to transients after the creation of a local model; (iv) uncertainty handling, in the sense of capability of dealing with different types of granular data; (v) higher stability and better convergence (if convergence is an issue); and (vi) higher computational speed and practical usability for model updates in order to increase the applicability and acceptability of evolving systems in real-world applications. Naturally, some of these more advanced issues have been addressed and investigated by the evolving systems community.

Because a significant part of the literature has been recently overviewed in Škrjanc et al. (2019), the next subsections focus on a more detailed formulation and review of the following methods:

– Multivariable Gaussian Evolving Participatory Learning (MG-ePL) (Lemos et al. 2011b);
– Evolving Cauchy Possibilistic Clustering (eCauchy) (Škrjanc et al. 2018);
– Evolving Granular Neural Network (eGNN) (Leite et al. 2013); and
– Evolving Fuzzy Linear Regression Tree (eFLRT) (Lemos et al. 2011a);

as examples of evolving frameworks of the distinct categories, as shown in Figure 1.

3 Formulation and algorithms of basic evolving frameworks

This section presents the formulation of the methods MG-ePL (Lemos et al. 2011b), eCauchy (Škrjanc et al. 2018), eGNN (Leite et al. 2013), and eFLRT (Lemos et al. 2011a), and summarizes the corresponding algorithms.

3.1 Multivariable Evolving Participatory Learning (MG-ePL)

A functional Takagi-Sugeno fuzzy model is composed by rules of the form:

R_i : If x_1 is A_{i1} and … and x_m is A_{im} then $y_i = a_{i0} + a_{i1}x_1 + \cdots + a_{im}x_m$

where R_i is the i-th fuzzy rule, for $i = 1, \dots, g^k$; g^k is the number of rules at step k; x_j, $j = 1, \dots, m$, is an input variable; A_{ij} are antecedent fuzzy sets; y_i is the local model output; and $a_{ij} \forall i, j$ are parameters associated to the consequent function of the i-th local model.

If the antecedent fuzzy sets are Gaussian, then:

$$\mu_{ij} = \exp\left(-\frac{4}{r^2}||x_j - x^*_{ij}||^2\right) \tag{1}$$

where r is the spread, ||.|| is the Euclidean norm, and x^*_i is the focal point, then the firing degree of a rule is computed using the product T-norm (Pedrycz and Gomide 2007):

$$\tau_i = \mu_{i1}(x_1) \times \mu_{i2}(x_2) \times \cdots \times \mu_{im}(x_m); \tag{2}$$

The overall model output is the weighted average of the local models:

$$y = \sum_{i=1}^{g^k} \lambda_i y_i \tag{3}$$

where $\lambda_i = \tau_i / \sum_{j=1}^{g^k} \tau_j$ is the normalized firing degree.

Unsupervised potential-based recursive clustering algorithms can be used to determine the fuzzy rules of the model. Clustering can be done in the input-output space in which each of the data points is given by $z = [x^T \ \ y]^T$. The existing clusters can be projected on the input-variables axes for interpretability purpose, see Figure 3.

The potential of a data sample z^k is a measure of its distance to all other data samples:

$$P(z^k) = \frac{1}{k-1}\sum_{i=1}^{k-1} \exp\left(-r||z^k - z^i||^2\right) \tag{4}$$

where $k = 2, 3, \dots$ is the index of processed data samples. The potential function finds data samples that can be considered centers of regions with higher data density, as shown in Fig. 4.

Recursive computation of potential, as developed in (Angelov and Filev 2004), is given as

$$P^k(z^*_l) = \frac{(k-1)P^{k-1}(z^*_l)}{k - 2 + P^{k-1}(z^*_l) + P^{k-1}(z^*_l)\sum_{j=1}^{m+1} d_j^{k(k-1)}} \tag{5}$$

where z^*_l is the center of the cluster l $(1 \times m + 1)$; and $d_j^{k(k-1)} = z^k_j - z^{k-1}_j$.

Coefficients of rule consequents are updated using the Recursive Weighted Least-Squares algorithm (Young 1984). Algorithm 1 summarizes the learning procedure of the evolving functional fuzzy model.

Algorithm 1 Evolving Functional Learning Algorithm

```
1: Compute the new data sample potential $P(z^k)$
2: for $j = 1, ..., g^k$ do
3:     Compute the center $c^j$ potential
4: end for
5: if $P(z^k) > P(c^j)\ \forall j$ then
6:     if $z^k$ is close enough to some cluster $j$ then
7:         $z^k$ replaces $c^j$ as the center of cluster $j$
8:     else
9:         A new cluster is created centered in $z^k$
10:    end if
11: else
12:    Update the consequent parameters of the closest cluster
13: end if
```

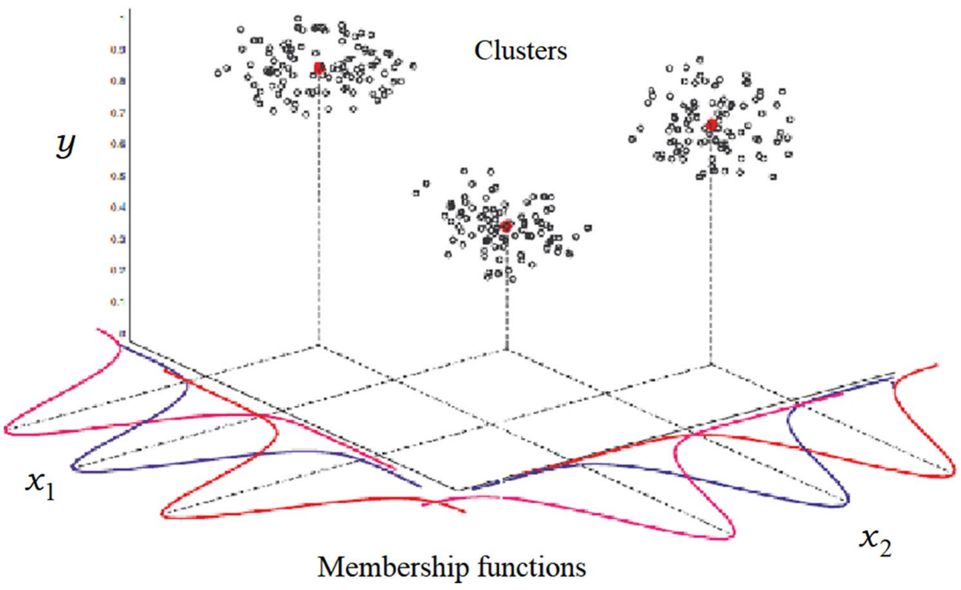


Fig. 3 Projection of fuzzy sets generated by clustering

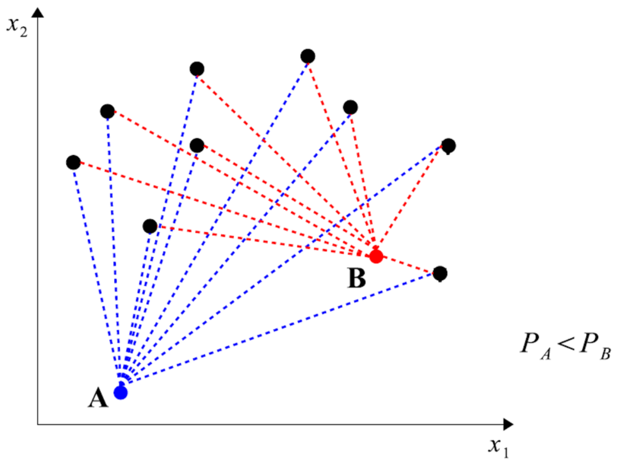


Fig. 4 Potential of data samples

An alternative evolving fuzzy model is the evolving multivariable Gaussian developed in Lemos et al. (2011b). The model uses a clustering algorithm derived from the participatory learning paradigm (Yager 1990). Different from previous neural and rule-based models (Kasabov and Song 2002; Lughofer 2008; Angelov and Filev 2004), the clustering procedure assumes that input variables may interact, and trigger ellipsoidal clusters whose axes are not necessarily parallel. Coefficients of the rule consequents, however, are also updated using Weighted Recursive Least Squares. The evolving multivariable Gaussian model avoids information loss (Abonyi et al. 2002).

Participatory learning assumes that the current knowledge about the system is part of the learning process itself and influences the way in which new observations are used in learning. It gives an automatic procedure to decide if a new observation lying far, outside the current cluster structure, is either a new cluster to be added to the model, or an outlier to be discarded. The cluster structure is updated using a compatibility measure $\rho_i^k \in [0, 1]$, and an arousal index, $a_i^k \in [0, 1]$. The compatibility measure computes how much an observation is compatible with the current cluster structure, while the arousal index acts as a critique to reveal when the current structure should be revised, given new input data.

Cluster centers are updated as follows:

$$v_i^{k+1} = v_i^k + G_i^k(x^k - v_i^k) \tag{6}$$

where G_i^k is defined as:

$$G_i^k = \beta(\rho_i^k)^{1-a_i^k} \tag{7}$$

and $\beta \in [0, 1]$ is the learning rate.

The arousal index gives a monitoring mechanism to observe the compatibility values. It is interpreted as the complement of the compatibility on the current knowledge. Figure 5 shows the participatory clustering procedure. The arousal index is computed using a sliding window assembled by the last w observations. It is viewed as the probability of observing less than a given number of violations of the compatibility threshold in a sequence of w observations. Algorithm 2 stands for the Gaussian Participatory evolving procedure.

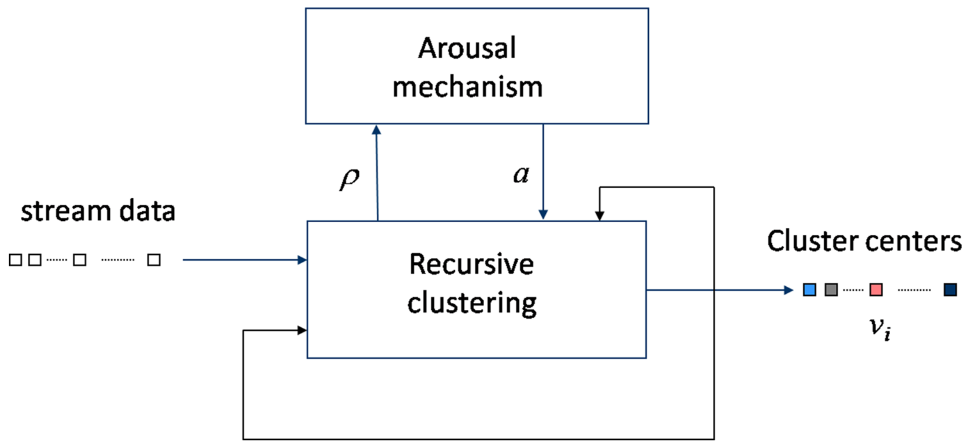


Fig. 5 Participatory clustering

Algorithm 2 Gaussian Participatory Evolving Clustering

```
1: Compute $\rho_i$ and $a_i$ for all clusters
2: Select the cluster with the highest compatibility ($j$)
3: if $\rho_i < T_\rho$ $\forall i$ and $a_j > T_a$ then
4:     Create a new cluster
5: else
6:     Update the parameters of cluster $j$
7: end if
8: Select the updated/created cluster ($idx$)
9: for all Clusters ($i$) do
10:     if Compatibility between $c_i$ and $c_{idx}$ is greater than $T_\rho$ then
11:         Merge redundant clusters
12:     end if
13: end for
```

Like its predecessors, the evolving multivariable Gaussian model is formed by functional fuzzy rules of the form:

$$R_i \quad : \quad \text{If} \quad x^k \quad \text{is} \quad H_i \quad \text{then} \quad y_i^k = \gamma_{io}^k + \sum_{j=1}^{m} \gamma_{ij}^k x_i^k \tag{8}$$

where R_i is the ith fuzzy rule, for $i = 1, \dots, c^k$; c^k is the number of rules, and γ_{io}^k and γ_{ij}^k are the consequent parameters at step k. Consequent parameters are updated using the Weighted Recursive Least Squares algorithm (Ljung 1999). The main steps of the learning algorithm are summarized in Algorithm 3.

3.2 Evolving Cauchy possibilistic clustering algorithm—eCauchy clustering

In this approach, proposed in Škrjanc et al. (2018), Škrjanc et al. (2019), Cauchy density is used to calculate membership of the data sample. The well-known possibilistic c-means clustering (PCM) is a special example of the proposed eCauchy algorithm. This approach can overcome the problems of modeling nonlinear data flows in highly noisy environments with frequent occurrence of outliers. The algorithm can be modified in different ways to solve different

Algorithm 3 Evolving Gaussian Multivariable Algorithm

```
1: Compute model output (estimation)
2: Update the cluster structure
3: if Cluster was created then
4:     Create a new rule
5: end if
6: if Cluster was modified then
7:     Update antecedent parameters of the respective rule using cluster parameters
8:     Update consequent parameters of the respective rule using weighted least squares
9: end if
10: if Two cluster were merged then
11:     Merge corresponding rules
12: end if
```

classification problems, and be used as a pre-treatment to solve regression problems. Due to the nature of the algorithm and easy computation, it is also suitable for solving big-data problems. The eCauchy algorithm described below needs only a few initial parameters, such as minimum and maximum density. The algorithm gradually changes the structure of the model based on the data stream, more precisely developing the structure of the model during the operation by adding, merging, splitting, and removing clusters. This approach allows the identification of very different clusters in terms of size and shape, and is insensitive to outliers and noise. The algorithm is explained in the following subsections.

3.2.1 Cauchy density for data stream

The Cauchy density of sample k for cluster j is defined as γ_k^j. Density is generally defined for data as the sum of distances between the current sample $z(k)$, of dimension $m \times 1$, and all previous samples belonging to a particular cluster (Angelov and Yager 2011; Blažič et al. 2014):

$$\gamma_k^j = \frac{1}{1 + \frac{1}{\sigma_l^2}\frac{\sum_{i=1}^{M^j} d_{ki}^j}{M^j}} \qquad j = 1, \dots, c, \tag{9}$$

where σ_l is a normalization constant that expands or narrows membership functions, d_{ki}^j denotes the square of the Euclidean distance between the current data sample $z(k)$ and the i-th sample from the j-th cluster z_i^j as follows:

$$d_{ki}^j = (z(k) - z_i^j)^T (z(k) - z_i^j) \tag{10}$$

and M^j indicates the number of samples in cluster j.

The generalization of the basic density measurement is introduced by positively defined internal matrix norm A^j with dimensions $m \times m$ as follows:

$$d_{ki}^j = (z(k) - z_i^j)^T A^j (z(k) - z_i^j) \tag{11}$$

Cauchy density becomes relative in this sense because the distances are weighed in different directions, with different weights. The equation for calculating the Cauchy density is then (Blažič et al. 2015):

$$\gamma_k^j = \frac{1}{1 + \frac{1}{\sigma_l^2}\frac{\sum_{i=1}^{M^j}(z(k)-z_i^j)^T A^j (z(k)-z_i^j)}{M^j}} \tag{12}$$

To use Cauchy density for on-line identification, the density equation (Eq. 12) must be converted to its recursive form Blažič et al. (2015):

$$\gamma_k^j = \frac{1}{1 + \frac{1}{\sigma_l^2}(z(k) - \mu^j)^T A^j (z(k) - \mu^j) + \frac{1}{\sigma_l^2} T_A^j}, \tag{13}$$

where $T_A^j = \frac{(M^j - 1)}{M^j} \operatorname{trace}(A^j \Sigma^j)$ and

$$\mu^j = \frac{1}{M^j}\sum_{i=1}^{M^j} z_i^j\,, \tag{14}$$

where μ^j defines the center of the j-th cluster of dimension $m \times 1$, and the lower and upper index at z_i^j define the i-th sample in the j-th cluster. The notation of cluster center can also be written as $\mu_{M_j}^j$ to explicitly express that the j-th cluster consists of M^j samples. Σ^j denotes the covariance matrix, of dimension $m \times m$, of the j-th cluster. The covariance matrix is calculated as:

$$\Sigma_{M^j}^j = \frac{1}{M^j - 1}\sum_{i=1}^{M^j}(z_i^j - \mu_{M^j}^j)(z_i^j - \mu_{M^j}^j)^T \tag{15}$$

The density, given by Eq. 13, is in its absolute form. The density value may be higher than one.

When the internal matrix norm is equal to the inverse covariance matrix $(\Sigma^j)^{-1}$ of the corresponding dataset, with data samples M^j, then the distance is called Mahalanobis distance. By introducing Eq. 11 with internal norm as the inverse of the covariance matrix in Eq. 9, we get a Cauchy-density relation based on the Mahalanobis distance (Blažič et al. 2014):

$$\gamma_k^j = \frac{1}{1 + \frac{1}{\sigma_l^2}(z(k) - \mu^j)^T (\Sigma^j)^{-1} (z(k) - \mu^j) + \frac{1}{\sigma_l^2}\frac{(M^j - 1)}{M^j} q}, \tag{16}$$

where q is the rank of the covariance matrix.

The advantage of using the Mahalanobis distance is that it describes a hyper-ellipsoid-shaped cluster. The size and shape of the hyper-ellipsoid depends on the covariance of the data within a cluster.

3.2.2 Recursive computation of Cauchy density

In an evolving algorithm, density must be calculated recursively. The proposed approach fully assigns the observed sample to the cluster with the highest density if it exceeds a predetermined minimum value. If the maximum sample density is lower than the threshold, then the sample is treated as an outlier (Škrjanc and Dovžan 2015) or as an initial sample of a new cluster (depending on the application).

With the new sample in the cluster $z(k)$, the number of samples increases, the mean and the cluster covariance matrix are updated in a recursive way. The update is performed in the following steps. First, calculate the difference between the current sample and the current mean:

$$e_{M^j}^j(k) = z(k) - \mu_{M^j}^j. \tag{17}$$

The centers of cluster is updated as follows

$$\mu^j_{M^j+1} = \mu^j_{M^j} + \frac{1}{M^j + 1} e^j_{M^j}(k). \tag{18}$$

and the states of the non-normalized covariance matrix are adapted as:

$$S^j_{M^j+1} = S^j_{M^j} + e^j_{M^j}(k)(z(k) - \boldsymbol{\mu}^j_{M^j+1})^T \tag{19}$$

The covariance matrix is then obtained as

$$\Sigma^j_{M^j+1} = \frac{1}{M^j} S^j_{M^j+1}. \tag{20}$$

The complete evolving Gaussian clustering algorithm is shown in Alg. 4.

3.2.3 Evolving Cauchy possibilistic clustering

The problems of data coming from the stream require evolving mechanisms such as adding, merging, and deleting clusters. A brief description of the underlying mechanisms within the eCauchy framework is given next.

3.2.4 Adding and removing clusters

For classification problems, the most common method is to use the direct adding method. Each sample is either added to one of the existing clusters, or a new cluster is initialized. In eCauchy, a new cluster is added if

$$\max_j \gamma^j_k < \Gamma_{max} \tag{21}$$

where Γ_{max} is a user-defined maximum-density threshold. By adding a new cluster, the number of clusters increases, $c = c + 1$, and the number of elements in the cluster is initialized to $M^j = 1$. The center and covariance of the cluster matrix are initialized to $\mu^j = z(k)$ and $\Sigma^j = 0$, respectively.

Algorithm 4 Algorithm of Evolving Cauchy Clustering.

```
1: Choice of $\Gamma_{max}$, $\sigma_l^2$
2: Initialization:
3:     $c \leftarrow 1$
4:     $M_1 \leftarrow 1$
5:     $\mu^1_{M_1} \leftarrow z(1)$
6:     $S^1 \leftarrow 0$
7: repeat $k \leftarrow k+1$, waiting for new sample $z(k)$
8:     Calculate densities $\gamma^i_k$, $i = 1, ..., c$
9:     Choice of maximal density $j = arg\max_i \gamma^j_k$
10:    if $\gamma^j_k \le \Gamma_{max}$
11:       Add and initialize new cluster
12:       $c \leftarrow c+1$
13:       $j = c$
14:       $M_j \leftarrow 1$
15:       $\mu^j_{M_j} \leftarrow z(k)$
16:       $S^j \leftarrow 0$
17:    else
18:       Update cluster $j$
19:       $M_j \leftarrow M_j + 1$
20:       $e_j(k) \leftarrow z(k) - \mu_j$
21:       $\mu^j_{M^j+1} = \mu^j_{M^j} + \frac{1}{M^j+1} e^j_{M^j}(k)$
22:       $S^j_{M^j+1} = S^j_{M^j} + e^j_{M^j}(k)(z(k) - \mu^j_{M^j+1})^T$
23:       $\Sigma^j_{M^j+1} = \frac{1}{M^j} S^j_{M^j+1}$
24:    end
25: until $k > N$
```

The purpose of the cluster removal mechanism is to detect clusters that have been added based on the outliers. The outliers often satisfy the adding condition. Such clusters are not valid. Since they were added based on outliers, they are unlikely to collect a noticeable number of samples over a predefined time period. Therefore, if the number of cluster samples is less than a given threshold, M_{min},

$$M^j < M_{min} \tag{22}$$

then the cluster is deleted (Dovžan et al. 2015; Angelov 2010).

The removing mechanism works well if the data is distributed and come to the learning algorithm more or less evenly from different classes or clusters. If the dataset is unbalanced, the removing mechanism can delete actual clusters, which may deteriorate the performance of the classification system. In other words, a sequence of samples of other classes (unbalanced case) may provide evidence that a specific inactive cluster should be deleted since it is useless in the current environment. Therefore, unbalanced data streams may require an additional approach to avoid the deletion of clusters that represent rare or seasonal events. However, outliers are generally rare events. In this case, inactive clusters, that is, clusters that were created based on outliers, should instead be deleted, which is a contradiction. This is an open issue in the evolving systems literature.

3.2.5 Merging and splitting clusters

As mentioned, adding clusters is based on the distance between the current sample and existing clusters. The distance or threshold to add a local model usually depends on the covariance of the data array around the center of the cluster. In some cases, especially when the samples arrive randomly from different classes and are very scattered, the learning algorithm usually generates more clusters than the necessary. A merging mechanism can be used to reduce the number of such clusters. Merging is a procedure that combines similar clusters. This reduces the number of clusters and simplifies the overall model structure. Cluster eligibility is usually determined by the Mahalanobis distance between two cluster centers; correlation between firing rates; or, in the case of regression models, based on information of the local models themselves (Dovžan et al. 2015). An innovative approach to merging is given in Škrjanc (2019).

The cluster splitting mechanism is designed for fine partitioning of the problem space. In the case of prediction problems, the algorithm splits clusters with large prediction errors. In the case of a classification problem, the algorithm splits clusters containing a certain number of misclassified samples. In this way, a more accurate partition of the problem space is obtained, and typically the model's accuracy increases (Dovžan et al. 2015). A detailed description of the cluster division is given in Lughofer et al. (2018).

3.3 Evolving granular neural network (eGNN)

Evolving Granular Neural Networks (eGNN) (Leite 2012; Leite et al. 2013) encode a set of fuzzy rules in their structures. Therefore, neural processing conforms with that of a fuzzy inference system. The network is supplied with fuzzy neurons, which perform aggregation functions (Beliakov et al. 2007); and with an incremental algorithm for learning from a data stream, which can be numerical or fuzzy. Fuzzy granules and rules are created gradually according to new information discovered from the numerical or fuzzy data. eGNN provides: (i) computational tractability and scalability with the number of samples and attributes; (ii) improved explainability and interpretability by means of granular local models and linguistic rules; and (iii) reduced cost of data processing in relation to non-evolving methods. eGNN has shown to be general, and able to outperform other evolving methods and models, including evolving classifiers and predictors (Leite et al. 2010; Leite 2012; Leite et al. 2013; Leite 2019).

Let $x = (x_1, \ldots, x_n)$ be an input vector and y be its corresponding output. Assume that the data stream $(x, y)^{[h]}$, $h = 1, \ldots,$ are samples measured from an unknown function f. Inputs x_j and output y can be symmetric fuzzy data in general—being interval and numerical data particular cases.

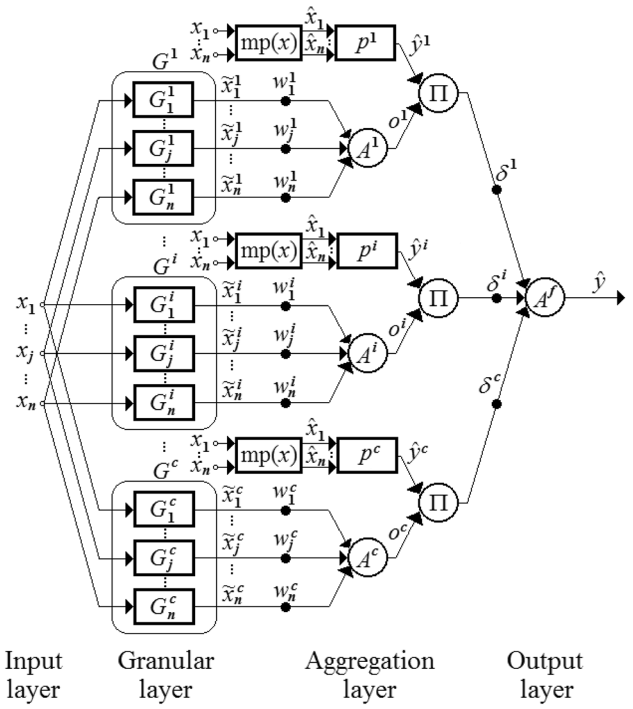


Fig. 6 eGNN topology and numerical output

Figure 6 depicts a four-layer eGNN structure. The first layer inputs samples $x^{[h]}$, one at a time, to the network. The granular layer consists of a collection of fuzzy sets G_j^i, $j = 1, \dots, n; i = 1, \dots, c$, stratified from the input data. Fuzzy sets G_j^i, $i = 1, \dots, c$, form a fuzzy partition of the j-th input domain, X_j. Similarly, fuzzy sets Γ^i, $i = 1, \dots, c$, give a fuzzy partition of the output domain Y. A granule $\gamma^i = G_1^i \times \cdots \times G_n^i \times \Gamma^i$ is a fuzzy relation, a multidimensional fuzzy set in $X_1 \times \cdots \times X_n \times Y$. Thus, granule γ^i has membership function $\gamma^i(x, y) = min\{G_1^i(x_1), \dots, G_n^i(x_n), \Gamma^i(y)\}$ in $X_1 \times \cdots \times X_n \times Y$. Granule γ^i is denoted by $\gamma^i = (G^i, \Gamma^i)$ with $G^i = (G_1^i, \dots, G_n^i)$, for short. Granule γ^i has a companion local function p^i. Real-valued affine functions:

$$p^i(\hat{x}_1, \dots, \hat{x}_n) = \hat{y}^i = a_0^i + \sum_{j=1}^{n} a_j^i \hat{x}_j, \tag{23}$$

are generally used. Parameters a_0^i and a_j^i are real values; $\hat{x}_j$ is the midpoint of $x_j = (\underline{\underline{x}}_j, \underline{x}_j, \overline{x}_j, \overline{\overline{x}}_j)$, computed as

$$\text{mp}(x_j) = \hat{x}_j = \frac{\underline{x}_j + \overline{x}_j}{2}. \tag{24}$$

Notice that $x_j = (\underline{\underline{x}}_j, \underline{x}_j, \overline{x}_j, \overline{\overline{x}}_j)$ is a trapezoidal membership function (a fuzzy datum), canonically represented by four parameters listed in ascending order. The intermediate parameters form the core, and the boundary parameters form its support.

Similarity degrees $\widetilde{x}^i = (\widetilde{x}_1^i, \dots, \widetilde{x}_n^i)$, see Fig. 6, is the result of matching between the input $x = (x_1, \dots, x_n)$ and fuzzy sets of $G^i = (G_1^i, \dots, G_n^i)$. As data and granules can be trapezoidal fuzzy objects, a useful similarity measure to quantify how the input data are related to the current knowledge is

$$\widetilde{x}_j^i = \begin{cases} 1 - \left(\dfrac{|\underline{\underline{g}}_j^i - \underline{\underline{x}}_j| + |\underline{g}_j^i - \underline{x}_j| + |\overline{g}_j^i - \overline{x}_j| + |\overline{\overline{g}}_j^i - \overline{\overline{x}}_j|}{4(\max(\overline{\overline{g}}_j^i, \overline{\overline{x}}_j) - \min(\underline{\underline{g}}_j^i, \underline{\underline{x}}_j))} \right) & \text{if } x_j \cap G_j^i \neq \emptyset \\ 0 & \text{otherwise.} \end{cases} \tag{25}$$

This measure returns $\widetilde{x}_j^i = 1$ for identical trapezoids, and reduces linearly as any numerator term increases. Non-overlapped trapezoids are considered dissimilar.

The aggregation layer is composed of fuzzy neurons A^i, $i = 1, \dots, c$, that combine the values from different inputs. In other words, a fuzzy neuron A^i combines weighted similarity degrees $(\widetilde{x}_1^i w_1^i, \dots, \widetilde{x}_n^i w_n^i)$ into a single value o^i. Notice that $\widetilde{x}_j^i$ is the result of matching between the j-th input attribute and the j-th trapezoidal membership function of the i-th granule, G_j^i, as in (25). An aggregation neuron, A^i, produces a diversity of nonlinear mappings between neuron inputs and output depending on the choice of weights w_j^i, $j = 1, \dots, n$, and aggregation function, e.g., triangular norms and conorms, null and uninorms, averaging and compensatory operators (Leite 2012; Leite et al. 2013).

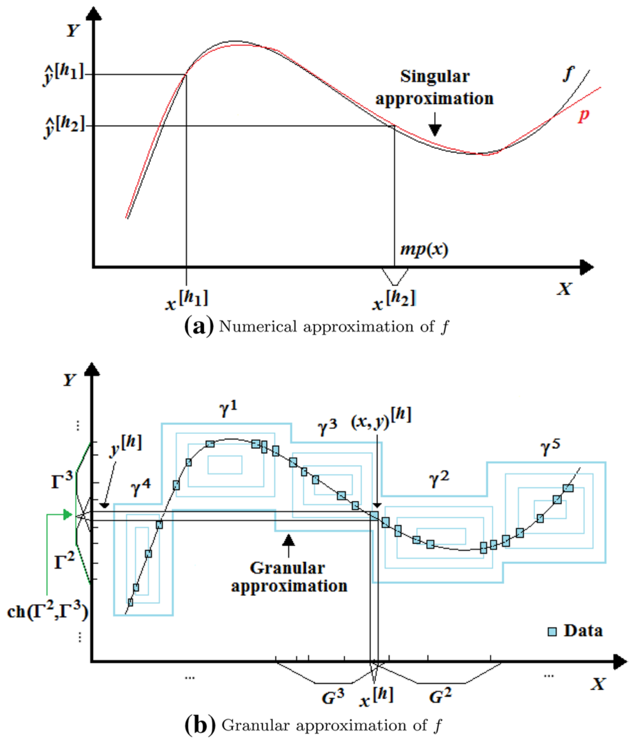


Fig. 7 eGNN numerical and granular approximation of function f

The output layer processes weighted values $(o^1\hat{y}^1\delta^1, \dots, o^c\hat{y}^c\delta^c)$ using a fuzzy neuron A^f to produce a numerical output $\hat{y}^{[h]}$.

An m-output eGNN needs a vector of local functions $(p_1^i, \dots, p_m^i)$, m output layer neurons $(A_1^f, \dots, A_m^f)$, and m outputs $(\hat{y}_1, \dots, \hat{y}_m)$. The network output $\hat{y}$, as shown in Figure 6, is a numerical approximation of f, independently if the input data are numerical or fuzzy.

Granular approximation of the function f at step H is given by a set of granules γ^i, $i = 1, \dots, c$, such that:

$$(x, y)^{[h]} \subseteq \bigcup_{i=1}^{c} \gamma^i, \; h = 1, \dots, H. \tag{26}$$

The granular approximation is constructed by granulating both, input data $x^{[h]}$ into fuzzy sets of G^i, and output data $y^{[h]}$ into fuzzy sets Γ^i. The granular approximation is the convex hull of output fuzzy sets Γ^{i^*}, where i^* are indices of active granules, that is, those for which $o^i > 0$. This guarantees that the numerical approximation $\hat{y}^{[h]}$ is enclosed by the granular approximation. The convex hull of trapezoidal fuzzy sets $\Gamma^1, \dots, \Gamma^i, \dots, \Gamma^c$, with $\Gamma^i = (\underline{\underline{u}}^i, \underline{u}^i, \overline{u}^i, \overline{\overline{u}}^i)$, is a trapezoidal fuzzy set whose representation is

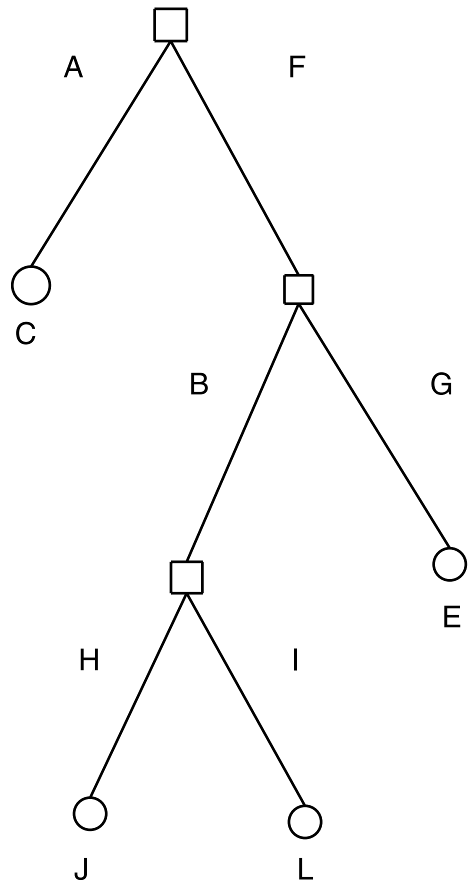


Fig. 8 Example regression tree

$$\mathrm{ch}(\Gamma^1,\ldots,\Gamma^c) = (min(\underline{\underline{u}}^1,\ldots,\underline{\underline{u}}^c), min(\underline{u}^1,\ldots,\underline{u}^c),$$
$$max(\overline{u}^1,\ldots,\overline{u}^c), max(\overline{\overline{u}}^1,\ldots,\overline{\overline{u}}^c)). \quad (27)$$

Particularly, the trapezoid $(\underline{\underline{u}}^{i^*}, \underline{u}^{i^*}, \overline{u}^{i^*}, \overline{\overline{u}}^{i^*})$, which results from $\mathrm{ch}(\Gamma^{i^*})$, $i^* = \{i : o^i > 0, i = 1,\ldots,c\}$, is a granular approximation of y. Note that granular approximation at step h does not depend on the availability of $y^{[h]}$ because o^i is obtained from $x^{[h]}$ (see Figure 6). Only the collection of output fuzzy sets Γ^i is required.

Figure 7 shows the numerical and granular approximation, p and $\bigcup_{i=1}^{c} \gamma^i$, of a function f. In Figure 7(a), a numerical input $x^{[h_1]}$ and a granular input $x^{[h_2]}$ produce numerical outputs $\hat{y}^{[h_1]}$ and $\hat{y}^{[h_2]}$ using p. In Figure 7(b), the granular input $x^{[h]}$ activates the fuzzy sets of G^2 and G^3. Therefore, the granular output is $\mathrm{ch}(\Gamma^2, \Gamma^3)$. Notice that $y^{[h]} \subset \mathrm{ch}(\Gamma^2, \Gamma^3)$.

eGNN develops simultaneously more precise (functional) and more interpretable (linguistic) fuzzy models. Accuracy and interpretability require tradeoffs and one usually excels over the other. Under assumption on specific weights and neurons types, fuzzy rules extracted from eGNN are of the type:

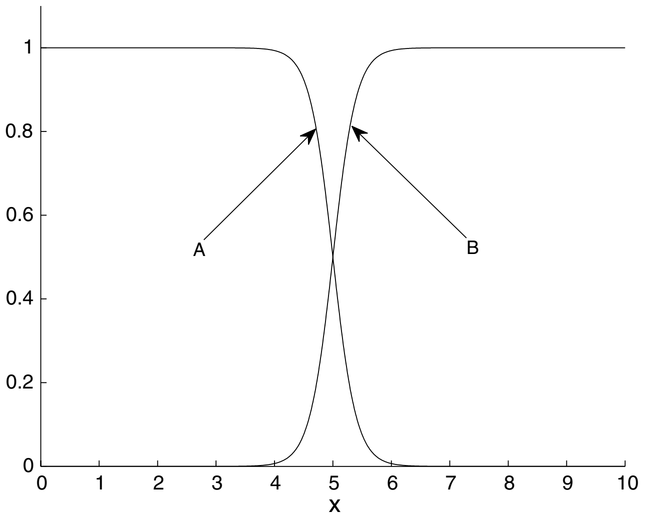


Fig. 9 Membership functions for *less than* and *greater than* 5

$$R^i : \text{ IF } (x_1 \text{ is } G_1^i) \text{ AND } \ldots \text{ AND } (x_n \text{ is } G_n^i)$$
$$\text{THEN } \underbrace{(\hat{y} \text{ is } \Gamma^i)}_{\text{linguistic}} \text{ AND } \underbrace{\hat{y} = p^i(x_1,\ldots,x_n)}_{\text{functional}}$$

Learning assumes that no granules and neurons exist *a priori*. Granules, neurons and connections can be added, updated, removed, and combined. Single pass over the data enables eGNN to address the issues of unbounded datasets and scalability. Fuzzy rules encoded in the eGNN structure aims to approximate and to enclose a target function. In summary, fuzzy rules and the network topology are obtained and updated incrementally to a new scenario from the Algorithm 5. See (Leite et al. 2013; Leite 2019) for details.

3.4 Evolving fuzzy linear regression tree (eFLRT)

Fuzzy regression trees replace binary splitting decisions at each tree node using pairs of membership functions, similarly as in fuzzy decision trees (Janikow 1998), and divide the input space in overlapping regions. Figure 8 shows an example of a regression tree.

There are several algorithms to grow linear regression trees and classic regression trees using incremental learning (Potts 2004 [38] Ikonomovska et al. 2009). While most evolving fuzzy-modeling methods use spatial information of the data space, the evolving fuzzy tree model granulates the data space selecting split points that improves the goodness of fit of the resulting models. Fuzzy regression trees replace splitting decision tests by two membership functions to mean *less than* and *greater than*. All branches of the tree fire in a degree. This results in an overlapping partition of the input space and a regression model based on a weighted

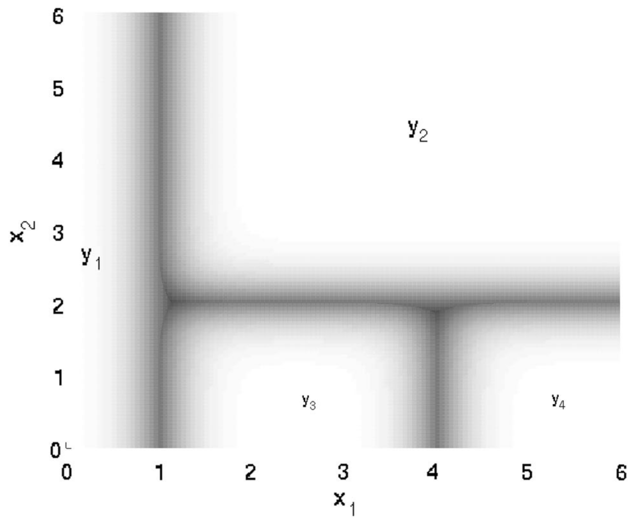


Fig. 10 Input space partition produced by a fuzzy regression tree

sum of local models. For instance, sigmoidal membership functions can be used in each internal node,

$$\mu(x) = \frac{1}{1 + \exp -\frac{1}{\sigma}(x - c)} \tag{28}$$

where c is the center, and σ is the spread. Figure 9 depicts membership functions describing *less than 5* and *greater than 5* for $\sigma = -0.5$ (*less than*) and $\sigma = 0.5$ (*greater than*).

Figure 10 shows the partition of the input space ($x_1 \in [0, 6]$ and $x_2 \in [0, 6]$) produced when the splitting tests of the tree illustrated in Figure 8 are replaced by pairs of sigmoidal membership functions, with c as the original split points of the figure, and $|\sigma| = 0.2$ for all internal nodes.

The tree computes the output as the weighted average of the output of all models assigned to the leaf nodes as follows:

$$\hat{y} = \frac{\sum_{i=1}^{l} y_i w_i}{\sum_{i=1}^{l} w_i} \tag{29}$$

where l is the number of leaves, and w_i is:

$$w_i = \mu_1(x)\ t\ \mu_2(x)\ t\ \ldots\ t\ \mu_o(x) = T_{j=1}^{o} \mu_j(x) \tag{30}$$

where t is a *t-norm*, o is the number of internal nodes reached from the root to the leaf i, and μ_j is one of the sigmoidal membership functions associated with internal node j.

Learning of evolving fuzzy regression trees is an incremental process that starts with a single-leaf tree and its corresponding regression model. The tree evolves as data are available by replacing leaves by subtrees using a recursive statistical model selection test. Parameters of the regression models of the leaves are updated using the Weighted Least-Squares algorithm. Algorithm 6 summarizes the learning algorithm. A detailed description of all evolving models and their application in distinct domains can be found in Lemos et al. (2013).

Algorithm 5 Evolving Granular Neural Network

```
1:  Select a type of neuron for the aggregation and output layers
2:  for all inputs $x^{[h]}$ do
3:      Compute compatibility degrees $(o^1, ..., o^c)$
4:      Aggregate values using $A^f$ to provide numerical approximation $\hat{y}^{[h]}$
5:      Compute convex hull of $\Gamma^{i^*}, i^* = \{i, o^i > 0\}$
6:      Find granular approximation $(\underline{\underline{u}}^{i^*}, \underline{u}^{i^*}, \overline{u}^{i^*}, \overline{\overline{u}}^{i^*})$
7:      Compute output error $\epsilon^{[h]} = y^{[h]} - \hat{y}^{[h]}$
8:      if $x^{[h]}$ is not within granules' expansion regions $E^i \forall i$ then
9:          Create granule $\gamma^{c+1}$, neuron $A^{c+1}$, and connections
10:     else
11:         Update the most active granule $\gamma^i$, $i = arg\ max(o^1, ..., o^c)$
12:         Adapt local function parameters $a^i_j$ using Recursive Least Squares
13:         Adapt weights $w^i_j \forall j, i$
14:     end if
15:     if $h = \alpha h_r$, $\alpha = 1, 2, ...$ then
16:         Combine granules when feasible
17:         Update model granularity $\rho$
18:         Adapt weights $\delta^i \forall i$
19:         Prune inactive granules and connections
20:     end if
21: end for
```

Algorithm 6 Evolving Fuzzy Regression Tree

```
 1: Compute the output and membership value of all leaves
 2: Update the linear models
 3: Select the leaf with the highest membership value
 4: for all inputs ($m$) do
 5:     for all candidate splits ($k$) do
 6:         Estimate the output replacing the selected leaf with the candidate split
 7:         Compute the *p-value* of the model selection test for the candidate split
 8:     end for
 9: end for
10: Select the candidate split associated with the minimum p-value
11: if p-value $< \frac{\alpha}{k \times m}$ then
12:     Replace the selected leaf by the candidate split
13: end if
```

4 Conclusion and future directions

This paper has introduced and recalled the main ideas and concepts of evolving intelligent systems. It gives an overview of the area with emphasis on the main system components, structures, categories, and learning algorithms that are part of the state-of-the-art of the area. Because of the steadily-increasing and large amounts of data witnessed in applications worldwide in different domains, it is safe to predict that soon evolving systems of different forms will become paramount to surpass current computational processing and storage challenges.

Although interesting and persuasive practical solutions have been achieved over the last sixteen years, future directions we see as essential for making evolving systems suitable to a broader field of applications, especially for Big data processing, Internet of Things (IoT), eXplainable Artificial Intelligence (XAI), Cyber-Physical Systems (CPS), and Smart Industry (SmI), are described in the following.

Propositions, lemmas, theorems and assurance that certain conditions are fulfilled are still lacking, in large part, in the field of evolving clustering and evolving neuro-fuzzy and rule-based modeling from data streams. For instance, necessary and sufficient conditions to guarantee short-term adaptation and long-term survivability are still to be found. This is a major challenge because it will require the formalization of concept shift and concept drift, and to show how they affect the hypothesis space from the point of view of simultaneous parameter estimation and structural adaptation. Systematic and formal approaches to deal with the stability-plasticity trade-off are still needed.

Missing data are very common in real-world applications. They arise due to incomplete observations, transfer problems, malfunction of sensors, or incomplete information obtained from experts or on public surveys. The missing-data issue is still an open topic in non-stationary data stream environments, in spite of having been extensively investigated in offline settings. Particular sequences of missing data may cause instability of Lyapunov-stable closed-loop control systems, and loss of memory in evolving intelligent models.

Characterization, design of experimental setups, and construction of workflows to guide development, performance evaluation, testing, validation, and comparison of algorithms in non-stationary environments require further elaboration. The evolution of rough-set models, Dempster-Shafer models, and aggregation functions are also important topics to expand the current scope of the area. T and S-norms, Uni and null-norms, and averaging functions are generally chosen *a priori* and kept fixed during model evolution. Approaches to switch aggregation operators based on properties of the data, and to update associated operator parameters are still to be undertaken.

Evolving systems in parallel high-performance computing frameworks will be explored in the next years. The rule-base modular and granular structure of fuzzy models is an interesting aspect to be exploited in high-frequency stream applications. Moreover, a variety of particularities of different applications and evolution aspects in hardware using low resources (aiming at smarter evolving models) are still to be addressed.

Funding This work was supported by Instituto Serrapilheira (Grant No. Serra-1812-26777), Javna Agencija za Raziskovalno Dejavnost RS (Grant No. P2-0219) and Conselho Nacional de Desenvolvimento Científico e Tecnológico (Grant No. 305906/2014-3).

References

Abonyi J, Babuška R, Szeifert F (2002) Modified gath-geva fuzzy clustering for identification of takagi-sugeno fuzzy models. IEEE Transactions on Systems Man and Cybernetics Part B (Cybernetics) 32(5):612–621

Agrawal R, Bala R (2008) Incremental bayesian classification for multivariate normal distribution data. Pattern Recognit Lett 29:1873–1876

Andonovski G, Angelov P, Blažič S, Škrjanc I (2016) A practical implementation of robust evolving cloud-based controller with normalized data space for heat-exchanger plant. Appl Soft Comput 48:29–38

Angelov PP (2002) Evolving rule-based models: a tool for design of flexible adaptive systems. Springer-Verlag, London

Angelov P (2012) Autonomous learning systems: from data streams to knowledge in real-time. Wiley, Hoboken

Angelov P, Filev D (2004) An approach to Online identification of Takagi–Suigeno fuzzy models. IEEE Trans Syst Man Cybern Part B-Cybern 34(1):484–498

Angelov P, Filev D, Kasabov N (2010) Evolving intelligent systems: methodology and applications. Wiley-IEEE Press, New Jersey

Angelov P, Kordon A (2010) Adaptive inferential sensors based on evolving fuzzy models. IEEE Trans Syst Man Cybern Part B Cybern 40(2):529–539

Angelov P, Lughofer E, Zhou X (2008) Evolving fuzzy classifiers using different model architectures. Fuzzy Sets Syst 159(23):3160–3182

Angelov P, Zhou X (2008) Evolving fuzzy-rule-based classifiers from data streams. IEEE Trans Fuzzy Syst 16:1462–1475

Angelov P, Ramezani R, Zhou X (2008) Autonomous novelty detection and object tracking in video streams using evolving clustering and takagi-sugeno type neuro-fuzzy system. In: IEEE international joint conference on neural networks (IJCNN), pp. 1456–1463

Angelov P, Yager R (2011) Simplified fuzzy rule-based systems using non-parametric antecedents and relative data density. In: 2011 IEEE Workshop on Evolving and Adaptive Intelligent Systems (EAIS), pp. 62–69

Angelov P, Zhou X (2006) Evolving fuzzy systems from data streams in real-time. In: 2006 International Symposium on Evolving Fuzzy Systems, pp. 29–35

Angelov P (2010) Evolving Intelligent Systems: Methodology and Applications, chap. Evolving Takagi-Sugeno Fuzzy Systems From Streaming Data (eTS+), pp. 21 – 50. Wiley, New Jersey

Angelov P (2010) Evolving takagi-sugeno fuzzy systems from streaming data, ets+. In: P. Angelov, D. Filev, N. Kasabov (eds.) Evolving Intelligent Systems: Methodology and Applications. Wiley-Interscience/IEEE Press

Angelov P, Gu X (2017) Mice: Multi-layer multi-model images classifier ensemble. In: Proceedings of the IEEE International Conference Cybernetics, pp. 1–8

Azeem MF, Hanmandlu H, Ahmad N (2003) Structure identification of generalized adaptive neurofuzzy inference systems. IEEE Trans Fuzzy Syst 11:666–681

Beliakov G, Pradera A, Calvo T (2007) Aggregation Functions: A Guide for Practitioners, 1st edn. Springer - Studies in Fuzziness and Soft Comput, Vol 21

Blažič S, Angelov P, Škrjanc I (2015) Comparison of approaches for identification of all-data cloud-based evolving systems. IFAC-PapersOnLine 48(10):129–134

Blažič S, Škrjanc I, Matko D (2014) A robust fuzzy adaptive law for evolving control systems. Evol Syst 5(1):3–10

Blažič S, Dovžan D, Škrjanc I (2014) Cloud-based identification of an evolving system with supervisory mechanisms. In: Proceedings of IEEE Control Systems Society Multiconference on Systems and Control, pp. 1906–1911

Bodyanskiy Y, Tyshchenko O, Kopaliani D (2016) Adaptive learning of an evolving cascade neo-fuzzy system in data stream mining tasks. Evol Syst 7(2):107–116

Bodyanskiy Y, Vynokurova O, Volkova V, Boiko O (2018) 2d-neo-fuzzy neuron and its adaptive learning. Inf Technol Manag Sci 21:24–28

Bordes A, Bottou L (2005) The huller: a simple and efficient online svm. In: Proceedings of the European Conf on Machine Learning. Springer, New York, pp. 505–512

Bordignon F, Gomide F (2014) Uninorm based evolving neural networks and approximation capabilities. Neurocomputing 127:13–20

Bueno L, Costa P, Mendes I, Cruz E, Leite D (2015) Evolving ensemble of fuzzy models for multivariate time series prediction. In: Proceedings of the IEEE Int Conf on Fuzzy Systems (FUZZ-IEEE), pp. 1–6

Doborjeh M, Kasabov N, Doborjeh ZG (2018) Evolving, dynamic clustering of spatio/spectro-temporal data in 3d spiking neural network models and a case study on eeg data. Evol Syst 9:195–211

Dovžan D, Logar V, Škrjanc I (2015) Implementation of an evolving fuzzy model (eFuMo) in a monitoring system for a waste-water treatment process. IEEE Trans Fuzzy Syst 23(5):1761–1776

Dovžan D, Škrjanc I (2011) Recursive clustering based on a Gustafson-Kessel algorithm. Evol Syst 2(1):15–24

Ferdaus M, Pratama M, Anavatti S, Garratt M (2019) Palm: An incremental construction of hyperplanes for data stream regression. IEEE Transactions on Fuzzy Systems, 15p., https://doi.org/10.1109/TFUZZ.2019.2893565

Filev D, Tseng F (2006) Novelty detection based machine health prognostics. In: 2006 International Symposium on Evolving Fuzzy Systems, pp. 193–199

Fritzke B (1994) Growing cell structures: a self-organizing network for unsupervised and supervised learning. Neural Netw 7:1441–1460

Garcia C, Leite D, Škrjanc I (2019) Incremental missing-data imputation for evolving fuzzy granular prediction. IEEE Transactions on Fuzzy Systems p. 15p. https://doi.org/10.1109/TFUZZ.2019.2935688

Hapfelmeier A, Pfahringer B, Kramer S (2014) Pruning incremental linear model trees with approximate lookahead. IEEE Trans Knowl Data Eng 26(8):2072–2076

Heeswijk M, Miche Y, Lindh-Knuutila T, Hilbers P, Honkela T, Oja E, Lendasse A (2009) Adaptive ensemble models of extreme learning machines for time series prediction. In: C. Alippi, P.C. Polycarpou M., G. Ellinas (eds.) Artificial Neural Networks - ICANN Lecture Notes in Computer Science. Springer - Berlin

Hisada M, Ozawa S, Zhang K, Kasabov N (2010) Incremental linear discriminant analysis for evolving feature spaces in multitask pattern recognition problems. Evol Syst 1:17–27

Iglesias JA, Ledezma A, Sanchis A (2014) An ensemble method based on evolving classifiers: estacking. In: Proceedings of the IEEE Symp on Evolving and Autonomous Learning Systems (EALS), pp. 1–8

Ikonomovska E, Gama J Learning model trees from data streams. In: J.F. Boulicaut, M. Berthold, T. Horváth (eds.) Discovery Science, Lecture Notes in Computer Science, vol. 5255, pp. 52–63. Springer Berlin, Heidelberg

Ikonomovska E, Gama J, Sebastião R, Gjorgjevik D (2009) Regression trees from data streams with drift detection. In: Proceedings of the 12th International Conference on Discovery Science, DS '09, pp. 121–135

Janikow C (1998) Fuzzy decision trees: issues and methods. IEEE Trans Syst Man Cybern Part B-Cybern 28(1):1–14

Kangin D, Angelov P, Iglesias JA, Sanchis A (2015) Evolving classifier tedaclass for big data. Procedia Comput Sci 53:9–18

Kasabov N (2007) Evolving connectionist systems: the knowledge engineering approach. Springer-Verlag, New York Inc, Secaucus

Kasabov N (2014) Neucube: a spiking neural network architecture for mapping, learning and understanding of spatio-temporal brain data. Neural Netw 52:62–76

Kasabov N, Song Q (2002) DENFIS: dynamic evolving neural-fuzzy inference system and its application for time-series prediction. IEEE Trans Fuzzy Syst 10(2):144–154

Klančar G, Škrjanc I (2015) Evolving principal component clustering with a low run-time complexity for LRF data mapping. Appl Soft Comput 35:349–358

Kolter J, Maloof M (2007) Dynamic weighted majority: an ensemble method for drifting concepts. J Mach Learn Res 8:2755–2790

Kwok TY, Yeung DY (1997) Constructive algorithms for structure learning in feedforward neural networks for regression problems. IEEE Trans Neural Netw 8(3):630–645

Leite D, Ballini R, Costa P, Gomide F (2012) Evolving fuzzy granular modeling from nonstationary fuzzy data streams. Evol Syst 3:65–79

Leite D, Costa P, Gomide F (2013) Evolving granular neural networks from fuzzy data streams. Neural Netw 38:1–16

Leite D, Costa P, Gomide F (2010) Granular approach for evolving system modeling. In: Hullermeier E, Kruse R, Hoffmann F (eds) Computational intelligence for knowledge-based systems design, vol 6178. Springer, Berlin - Heidelberg, pp 340–349 Lecture Notes in Computer Science

Leite D, Costa P, Gomide F (2012) Interval approach for evolving granular system modeling. In: Sayed-Mouchaweh M, Lughofer E (eds) Learning in non-stationary environments. Springer, New York, pp 271–300

Leite D, Palhares R, Campos V, Gomide F (2015) Evolving granular fuzzy model-based control of nonlinear dynamic systems. IEEE Trans Fuzzy Syst 23:923–938

Leite D, Škrjanc I (2019) Ensemble of evolving optimal granular experts, owa aggregation, and time series prediction. Inf Sci 504:95–112

Leite D (2012) Evolving granular systems. Ph.D. thesis, University of Campinas, School of Electrical and Computer Engineering

Leite D (2019) Comparison of genetic and incremental learning methods for neural network-based electrical machine fault detection. In: E. Lughofer, M. Sayed-Mouchaweh (eds.) Predictive Maintenance in Dynamic Systems, pp. 231–268. Springer - Cham

Leite D, Andonovski G, Škrjanc I, Gomide F (2019) Optimal rule-based granular systems from data streams. IEEE Transactions on Fuzzy Systems, 14p., https://doi.org/10.1109/TFUZZ.2019.2911493

Leite D, Costa P, Gomide F (2010) Evolving granular neural network for semi-supervised data stream classification. In: International Joint Conference on Neural Networks (IJCNN), pp. 1–8

Leite D, Santana M, Borges A, Gomide F (2016) Fuzzy granular neural network for incremental modeling of nonlinear chaotic systems. In: Proceedings of the IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 64–71

Lemos A, Caminhas W, Gomide F (2010) Fuzzy multivariable gaussian evolving approach for fault detection and diagnosis. In: Hullermeier E, Kruse R, Hoffmann F (eds) Computational intelligence for knowledge-based systems design, vol 6178. Springer, Berlin / Heidelberg, pp 360–369 Lecture Notes in Computer Science

Lemos A, Caminhas W, Gomide F (2013) Evolving intelligent systems: methods, algorithms and applications. In: Ramanna S, Jain L, Howlett R (eds) Emerging paradigms in machine learning. Springer, New York

Lemos A, Gomide F, Caminhas W (2011) Fuzzy evolving linear regression trees. Evol Syst 2(1):1–14

Lemos A, Gomide F, Caminhas W (2011) Multivariable gaussian evolving fuzzy modeling system. IEEE Trans Fuzzy Syst 19(1):91–104

Leng G, Prasad G, McGinnty TM (2004) An online algorithm for creating self-organizing fuzzy neural networks. Neural Netw 17:1477–1493

Lima E, Hell M, Ballini R, Gomide F (2010) Evolving fuzzy modeling using participatory learning. In: P. Angelov, D. Filev, N. Kasabov (eds.) Evolving intelligent systems: methodology and Applications. Wiley-Interscience/IEEE Press

Ljung L (1999) System Identification. Prentice-Hall, Upper Saddle River

Lughofer E (2008) Extensions of vector quantization for incremental clustering. Pattern Recognit 41(3):995–1011

Lughofer ED (2008) FLEXFIS: a robust incremental learning approach for evolving Takagi-Sugeno fuzzy models. IEEE Trans Fuzzy Syst 16(6):1393–1410

Lughofer E, Bouchot JL, Shaker A (2011) Online elimination of local redundancies in evolving fuzzy systems. Evol Syst 2:165–187

Lughofer E, Buchtala O (2013) Reliable all-pairs evolving fuzzy classifiers. IEEE Trans Fuzzy Syst 21:625–641

Lughofer E, Cernuda C, Kindermann S, Pratama M (2015) Generalized smart evolving fuzzy systems. Evol Syst 6:269–292

Lughofer E, Pratama M, Škrjanc I (2018) Incremental rule splitting in generalized evolving fuzzy systems for autonomous drift compensation. IEEE Trans Fuzzy Syst 26(4):1854–1865

Lughofer E (2011) Evolving Fuzzy Systems: Methodologies, Advanced Concepts and Applications, vol. 266. Studies in Fuzziness and Soft Computing Series, J. Kacprzyk (Ed.), Springer-Verlag, Berlin Heidelberg

Maciel L, Ballini R, Gomide F (2017) An evolving possibilistic fuzzy modeling approach for value-at-risk estimation. Appl Soft Comput 60:820–830

Maciel L, Ballini R, Gomide F (2018) Evolving fuzzy modelling for yield curve forecasting. Int J Econ Bus Res 15:290–311

Malcangi M, Grew P (2017) Evolving connectionist method for adaptive audiovisual speech recognition. Evol Syst 8:85–94

Malcangi M, Quan H, Vaini E, Lombardi P, Rienzo M (2018) Evolving fuzzy-neural paradigm applied to the recognition and removal of artefactual beats in continuous seismocardiogram recordings. Evol Syst, 10p., https://doi.org/10.1007/s12530-018-9238-8

Pedrycz W, Gomide F (2007) Fuzzy systems engineering: toward human-centric computing. Wiley Interscience, NJ

Polikar R (2006) Ensemble based systems in decision making. IEEE Circuits Syst Mag 6:21–45

Potts D (2004) Incremental learning of linear model trees. In: ICML '04: Proceedings of the twenty-first international conference on Machine learning, p. 84. ACM, New York, NY, USA

Prasad M, Za'in C, Pratama M, Lughofer E, Ferdaus M, Cai Q (2018) Big data analytics based on panfis mapreduce. Procedia Comput Sci 144:140–152

Pratama M, Anavatti S, Angelov P, Lughofer E (2014) Panfis: a novel incremental learning machine. IEEE Trans Neural Netw Learn Syst 25:55–67

Pratama M, Anavatti S, Lu J (2015) Recurrent classifier based on an incremental meta-cognitive scaffolding algorithm. IEEE Trans Fuzzy Syst 23:2048–2066

Pratama M, Lu J, Lughofer E, Zhang G, Er M (2016) An incremental learning of concept drifts using evolving type-2 recurrent fuzzy neural networks. IEEE Trans Fuzzy Syst 25:1175–1192

Pratama M, Pedrycz W, Lughofer E (2018) Evolving ensemble fuzzy classifier. IEEE Trans Fuzzy Syst 26:2552–2567

Pratama M, Wang D (2019) Deep stacked stochastic configuration networks for lifelong learning of non-stationary data streams. Inf Sci 495:150–174

Precup RE, Teban TA, Albu A, Szedlak-Stinean AI, Bojan-Dragos CA (2018) Experiments in incremental online identification of fuzzy models of finger dynamics. Rom J Inf Sci Technol 21:358–376

Rong HJ, Sundararajan N, Huang GB, Zhao GS (2011) Extended sequential adaptive fuzzy inference system for classification problems. Evol Syst 2:71–82

Rubio JJ (2009) Sofmls: online self-organizing fuzzy modified least square network. IEEE Trans Fuzzy Syst 17:1296–1309

Rubio JJ (2014) Evolving intelligent algorithms for the modelling of brain and eye signals. Appl Soft Comput 14:259–268

Rubio JJ (2017) Usnfis: uniform stable neuro fuzzy inference system. Neurocomputing 262:57–66

Rubio JJ, Bouchachia A (2017) Msafis: an evolving fuzzy inference system. Soft Comput 21:2357–2366

Silva A, Caminhas W, Lemos A, Gomide F (2013) A fast learning algorithm for evolving neo-fuzzy neuron. Appl Soft Comput 14:194–209

Silva A, Caminhas W, Lemos A, Gomide F (2015) Adaptive input selection and evolving neural fuzzy networks modeling. Int J Comput Intell Syst 8:3–14

Silva S, Costa P, Gouvea M, Lacerda A, Alves F, Leite D (2018) High impedance fault detection in power distribution systems using wavelet transform and evolving neural network. Electr Power Syst Res 154:474–483

Silva S, Costa P, Santana M, Leite D (2018) Evolving neuro-fuzzy network for real-time high impedance fault detection and classification. Neural Comput & Applic, 14p., https://doi.org/10.1007/s00521-018-3789-2

Soares E, Costa P, Costa B, Leite D (2018) Ensemble of evolving data clouds and fuzzy models for weather time series prediction. Appl Soft Comput 64:445–453

Soares E, Camargo H, Camargo S, Leite D (2018) Incremental gaussian granular fuzzy modeling applied to hurricane track forecasting. In: IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 1–8

Soleimani-B H, Lucas C, Araabi BN (2010) Recursive gath-geva clustering as a basis for evolving neuro-fuzzy modeling. Evol Syst 1:59–71

Takagi T, Sugeno M (1985) Fuzzy identification of systems and its applications to modeling and control. IEEE Trans Syst Man Cybern 15(1):116–132

Tung S, Quek C, Guan C (2013) et2fis: an evolving type-2 neural fuzzy inference system. Inf Sci 220:124–148

Tzafestas SG, Zikidis KC (2001) Neurofast: on-line neuro-fuzzy art-based structure and parameter learning tsk model. IEEE Trans Syst Man Cybern— Part B 31:797–802

Wang W, Vrbanek J (2008) An evolving fuzzy predictor for industrial applications. IEEE Trans Fuzzy Syst 16(6):1439–1449

Williamson JR (1996) Gaussian ARTMAP: a neural network for past incremental learning of noisy multidimensional maps. Neural Netw 9(5):881–897

Wu S, Er MJ (2000) Dynamic fuzzy neural networks—a novel approach to function approximation. IEEE Trans Syst Man Cybern—Part B 30:358–364

Wu S, Er MJ, Gao Y (2001) A fast approach for automatic generation of fuzzy rules by generalized dynamicc fuzzy neural networks. IEEE Trans Fuzzy Syst 9:578–594

Yager R (1990) A model of participatory learning. IEEE Trans Syst Man Cybern 20(5):1229–1234

Young P (1984) Recursive estimation and time-series analysis: an introduction. Springer-Verlag, New York Inc, New York

Yourdshahi ES, Angelov P, Marcolino L, Tsianakas G (2018) Towards evolving cooperative mapping for large-scale uav teams. In: Proceedings of the IEEE Symposium Series on Computational Intelligence (SSCI), pp. 2262–2269

Za'in C, Pratama M, Lughofer E, Anavatti S (2017) Evolving type-2 web news mining. Appl Soft Comput 54:200–220

Zdešar A, Dovžan D, Škrjanc I (2014) Self-tuning of 2 DOF control based on evolving fuzzy model. Appl Soft Comput 19:403–418

Škrjanc I (2009) Confidence interval of fuzzy models: an example using a waste-water treatment plant. Chemom Intell Lab Syst 96:182–187

Škrjanc I (2015) Evolving fuzzy-model-based design of experiments with supervised hierarchical clustering. IEEE Trans Fuzzy Syst 23(4):861–871

Škrjanc I, Andonovski G, Ledezma A, Sipele O, Iglesias JA, Sanchis A (2018) Evolving cloud-based system for the recognition of drivers' actions. Expert Syst Appl 99:231–238

Škrjanc I, Blažič S, Lughofer E, Dovžan D (2019) Inner matrix norms in evolving Cauchy possibilistic clustering for classification and regression from data streams. Inf Sci 478:540–563

Škrjanc I, Dovžan D (2015) Evolving Gustafson-Kessel possibilistic c-means clustering. Procedia Comput Sci 53:191–198

Škrjanc I, Iglesias JA, Sanchis A, Leite D, Lughofer E, Gomide F (2019) Evolving fuzzy and neuro-fuzzy approaches in clustering, regression, identification, and classification: a survey. Inf Sci 490:344–368

Škrjanc I, Ozawa S, Ban T, Dovžan D (2018) Large-scale cyber attacks monitoring using evolving cauchy possibilistic clustering. Appl Soft Comput 62:592–601

Škrjanc I (2019) Cluster-volume-based merging approach for incrementally evolving fuzzy Gaussian clustering - eGAUSS+. IEEE Trans Fuzzy Syst pp. 1–11. https://doi.org/10.1109/TFUZZ.2019.2931874