Incremental Learning of Concept Drift Using Evolving Type-2 Recurrent Fuzzy Neural Network

Mahardhika Pratama, Jie Lu, Edwin Lughofer, Guangquan Zhang, Meng Joo Er

Abstract— the age of big data and dynamic environments result in the increasing demand of advanced machine learning techniques to deal with concept drifts in large data streams. Evolving Fuzzy Systems (EFS) are one of recent initiatives from the fuzzy system community to resolve the issue. Existing EFSs are not robust against data uncertainty, temporal system dynamics, and the absence of system order, because vast majority of EFSs are designed in the feed-forward type-1 fuzzy network architecture. This paper aims to solve the issue of data uncertainty, temporal behaviour, and the absence of system order by developing a novel evolving recurrent fuzzy neural network, called Evolving Type-2 Recurrent Fuzzy Neural Network (eT2RFNN). eT2RFNN is constructed in a new recurrent network architecture, featuring double recurrent layers. The new recurrent network architecture evolves a generalized interval type-2 fuzzy rule, where the rule premise is built upon the interval type-2 multivariate Gaussian function, while the rule consequent is crafted by the non-linear wavelet function. The eT2RFNN adopts a holistic concept of evolving systems, where the fuzzy rule can be automatically generated, pruned, merged and recalled in the single pass learning mode. eT2RFNN is capable of coping with the problem of high dimensionality, because it is equipped with online feature selection technology. The efficacy of eT2RFNN has been experimentally validated using artificial and real-world data streams and compared with prominent learning algorithms. eT2RFNN can produce more reliable predictive accuracy, while retaining lower complexity than its counterparts.

Index Terms—evolving fuzzy systems, fuzzy neural networks, recurrent fuzzy neural networks, type-2 fuzzy systems, incremental learning, concept drifts

I. INTRODUCTION

Large data streams and dynamic environments are the two most challenging research problems in today's real-world big data applications [20],[44],[45],[55]. Every day, a massive amount of data streams are generated from sensors, the internet, etc. These data streams usually do not follow static and predictable data distributions, because they are influenced by rapidly changing statistical parameters. The Evolving Fuzzy System (EFS) [1] has gained increasing popularity in the machine learning community, because it answers an urgent demand for 'learning from non-stationary data streams'. EFS answers desiderata for data stream mining for four reasons: 1) EFS works in the

truly sequential (or one-pass) learning scenario, which is capable of processing large data streams on the fly [2]; 2) EFS is capable of self-organising fuzzy rules in accordance with its relevance to the current learning environment [46]; 3) EFS has a built-in rule growing procedure, which is capable of automatically generating a new fuzzy rule when encountering new knowledge [3],[47]. This learning module is equivalent to a drift detection scenario in conventional machine learning, because it monitors the conflict level induced by a data stream;4) EFS is equipped by rule base simplification modules: rule merging, pruning, etc, which warrants the rule base complexity at a low level and avoids the overfitting case. EFS deserves more in-depth investigation, because existing EFSs are mostly built upon the type-1 fuzzy system, which features a crisp and certain characteristic [4]. This makes the EFS not robust to cope with information uncertainty as a result of noisy measurement, noisy data, and disagreement of expert knowledge [5]. Furthermore, vast majority of the EFS are constructed in the feed-forward network architecture, which cannot properly handle the temporal system dynamic and requires system order to be known in order for the I/O relationship of the regression model to be defined.

We propose a novel Evolving Type-2 Recurrent Fuzzy Neural Network (eT2RFNN), which is capable of resolving all three underlying issues in an online mode: **uncertainty**, **temporal system dynamic**, **and system order**. eT2RFNN constitutes a fully evolving and adaptive learning algorithm, that can initiate its learning process from scratch with an empty rule base and automatically add, prune, merge its fuzzy rules from data streams afterwards. eT2RFNN borrows three learning modules proposed in [17] : 1) The rule growing process is governed by the Type-2 Data Quality (T2DQ) method; 2) the parameter learning scenario relies on the Fuzzily Weighted Generalized Recursive Least Square (FWGRLS) method; 3) the rule merging process is done by virtue of the Extended Vector Similarity (EVS) concept. Meanwhile, eT2RFNN conveys four novel learning components as follows:

- *A Novel Recurrent Network Architecture:* eT2RFNN puts forward a new recurrent network topology, which presents a double local recurrent connection at both rule layer [12] and consequent layer [18]. The recurrent connection at the rule layer functions to delineate the temporal firing

strength of the model by memorising the membership degree of the previous sample, whereas the recurrent link at the consequent layer aims to expedite the dynamic of the Wavelet function. To the best our knowledge, existing recurrent fuzzy neural networks still make use of a single recurrent layer in the form of local, global or interactive feedback.

- *A Generalized Interval Type-2 Fuzzy Rule:* A generalised interval type-2 fuzzy rule is proposed in this paper, which employs a multivariate Gaussian function in the input part and utilises a nonlinear Wavelet function in the output part. The main difference between this rule and that of [17] can be seen in the rule consequent, where the nonlinear Wavelet function is exploited. This amendment is to address the demanding memory demand of the Chebyshev polynomial [18]. This fuzzy rule can be perceived as an extension of [10], because it triggers arbitrarily rotated ellipsoidal rule instead of the ellipsoidal rule in the main axes.
- *Type-2 Relative Mutual Information (T2RMI) method*: A novel rule pruning method, namely the T2RMI method, is proposed. The T2RMI method is capable of detecting inconsequential fuzzy rules by scrutinising the correlation between the fuzzy rules and the target variables. The original version of the RMI method was originally initiated in [18]. In this paper, the T2RMI method is extended to suit the working principle of a type-2 fuzzy system. Furthermore, the T2RMI method adopt a different concept of the utility method [20], popular in the literature. The significance of a rule is derived from the issue of relevance of a rule to a target variable, whereas the utility method relies on the percentage of usage of a rule during its lifespan.
- *Sequential Markov Blanket Criterion (SMBC)*: Several online feature selection scenarios have been designed for EFS [20], [21],[48]. All of which are based on the relevance of input attributes, and discount possible redundancy issues among input attributes. In this paper, a novel online feature selection, namely SMBC, is put forward, which depicts the online version of MBC [22]. The SMBC takes into account both relevance and redundancy facets in determining the importance of input features.

To sum up, this paper proposes a solution to address the issue of **uncertainty**, **temporal system dynamic**, **absence of system order** simultaneously in learning from large data streams. Major

contribution of this paper can be found in four aspects:1) a generalized recurrent network architecture with double self-feedback loops at rule and consequent layers are put forward, 2) a new type of the interval type-2 fuzzy rule is presented, where the rule premise is crafted by the interval type-2 multivariate Gaussian function with uncertain means, while the rule consequent is driven by the nonlinear Wavelet function, 3) this paper also offers two novel learning components of the EFS, the T2RMI method for the rule pruning process and the SMBC method for the online feature selection mechanism, 4) The universal approximation capability of eT2RFNN is also investigated. It is well-known that the type-2 fuzzy system is essentialy a universal approximator, because it is a modified version of the type-1 fuzzy system. An open issue is, however, in the case of the recurrent fuzzy neural network, whose inter-temporal dependency is to the best of our knowledge uncharted in a standard universal approximation theorem of the fuzzy neural network. This paper discusses the universal approximation capability of the eT2RFNN by extending the result of [59] for the single recurrent connection architecture to the double recurrent connection architecture. The efficacy of the eT2RFNN is experimentally validated using various real-world and synthetic data streams and are also benchmarked with state-of-the art EFSs, where eT2RFNN demonstrates the most encouraging numerical results in both complexity and accuracy. The remainder of this paper is organised as follows: Section 2 deliberates the network architecture of eT2RFNN; Section 3 elaborates the learning policy of eT2RFNN and also the complexity analysis; Section 4 details the numerical studies and the experiments based comparisons with its counterparts; Conclusions are drawn in Section 5.

II. Related Works

A. Type-2 Fuzzy Systems

Real-world data, by their nature, contain bias, noise, abnormalities due to faulty sensors, etc., resulting in a high degree of uncertainty. Because training data do not truly represent system dynamic, an accurate paremeter identification strategy as necessitated in the type-1 fuzzy system becomes an extremely difficult task. In light of this issue, The type-2 fuzzy system [5] characterises a fuzzy membership via a so-called fuzzy-fuzzy set, which provides some tolerance against information uncertainty. The issue of complexity hinders the viability of the type-2 fuzzy system

in online real-time situations. In addition, it cannot be handled by well-known type-1 fuzzy mathematics. This drawback has led to the notion of the interval type-2 fuzzy system [6], which presents a simplified version of the type-2 fuzzy system. The interval type-2 fuzzy system assumes the secondary grade of a membership function as unity to enable the use of the type-1 fuzzy mathematics. Three configurations of non-evolving interval type-2 fuzzy system and their learning algorithms were put forward in [50]. The interval type-2 fuzzy system concept has been implemented in an evolving network structure in [7]. However, the initial version of the interval type-2 fuzzy neural network is overly dependent on the Karnik-Mendel (KM) type reduction method, computationaly prohibitive, because the rule consequent has to be first reordered in an ascending manner and then the KM method is carried out to find the crossover points [8]. To this end, the construct of a q coefficient was proposed in [9] to perform the type reduction scenario in lieu of the KM method. This method has been incorporated into the scope of EFS in [10]. The concept of interval type-2 fuzzy system has gained tremendous success in various real-world applications: edge detection [51], time-series prediction [52], intelligent agent in ambient environments [53], congestion control in the video streaming across IP networks [54]. However, the interval type-2 EFS is still an open research area because of three technical flaws: 1) it is constructed in the feed-forward network architecture, which cannot properly model temporal behaviour and is sensitive to the number of delayed input attributes; 2) it endures the absence of a rule base simplification method, which may result in costly pre- and/or post-training steps; 3) vast majority of existing works have not incorporated an online feature selection scenario and thus usually treat feature selection as a pre-processing step.

B. Recurrent Neural Networks

In contrast to its feedforward counterpart, the recurrent network architecture is fitted out by an internal memory component in the form of a feedback or recurrent layer [11], which offers a plausible solution to surmount the temporal property of the data streams. This component can store the previous characteristics of the system dynamic, and is thereby able to handle the temporal system dynamic reliably. This appealing trait is also capable of overcoming the absence of system order, because the internal memory component can feed the past behaviour of the system in lieu

of the lagged input and/or output variables in the I/O relationship of the regression model. In general, recurrent fuzzy neural networks can be classified into three types with respect to their recurrent layers: local [12], global [13], and interactive [14]. In our opinion, the local recurrent layer is the most suitable one for the EFS, because it imposes the lowest complexity and is applicable in the local learning context [15]. Recently, the notion of the recurrent network topology has been integrated into the realm of the interval type-2 fuzzy neural network in [16]. These works, however, rely on the KM iterative procedure, which is computationally expensive Table 1 summarises key characteristics of 14 related works, reviewed in Section 1 and 2.

Table 1. Main Features of Related Works

ALGORITHMS	REFERENCES	Type	Structure	Learning Principle
eTS	[1]	Type-1 Fuzzy	Feedforward	Evolving
Simp_eTS	[2]	Type-1 Fuzzy	Feedforward	Evolving
PANFIS	[3]	Type-1 Fuzzy	Feedforward	Evolving
SEIT2FNN	[7]	Interval Type-2 Fuzzy	Feedforward	Evolving
T2TSKFNS	[9]	Interval Type-2 Fuzzy	Feedforward	Static
TSCIT2FNN	[10]	Interval Type-2 Fuzzy	Feeforward	Evolving
eT2Class	[17]	Interval Type-2 Fuzzy	Feedforward	Evolving
IT2FNN	[50]	Interval Type-2 Fuzzy	Feedforward	Static
RSONFIN	[11]	Type-1 Fuzzy	Single Globally Recurrent	Evolving
RSEFNN-LF	[12]	Type-1 Fuzzy	Single Locally Recurrent	Evolving
TRFN	[13]	Type-1 Fuzzy	Single Globally Recurrect	Evolutionary
IRSFNN	[14]	Type-1 Fuzzy	Single Interactively Recurrent	Evolving
rClass	[15]	Type-1 Fuzzy	Single Locally Recurrent	Evolving
RSEIT2FNN	[16]	Interval Type-2 Fuzzy	Single Locally Recurrent	Evolving

III. Network Architecture of eT2RFNN

In this section, the network architecture of eT2RFNN is outlined. The eT2RFNN puts forward a novel recurrent network architecture to produce its final output. The unique property of the network topology lies on double local recurrent links at both rule and consequent layers. In essence, the recurrent property allows a model to store previous information, thus being more robust to temporal system dynamics [11]. It also resolves the bottleneck in ascertaining the system order of the model. It is worth noting that, in the feed-forward fuzzy neural network, the output is often the function of previous input and/or output. On the other hand, the use of the wavelet function in the rule consequent can retard the reaction speed against rapidly changing data distributions. The decline is compensated for by the recurrent connection in the rule consequent. Several variants of recurrent networks have been published: interactive [14], global [13], and local [12]. The local recurrent connection is utilised in the eT2RFNN, because it still retains the local learning trait, where each fuzzy rule is trained separately and the learning process of one rule has

no effect on the stability of other rules. It is worth mentioning that a single recurrect connection is usually enough to induce an internal memory component property of the network architecture. The eT2RFNN is , however, equipped with the second recurrent link in the consequent layer to further strengthen robustness against the temporal problem and importantly to increase sensitivity of the Wavelet function against changing operating conditions [18]. Fig.1 shows the network architecture of the ET2RFNN.

The network architecture generates a generalized interval type-2 fuzzy rule. It utilises the interval type-2 multivariate Gaussian function with uncertain means in the hidden layer, and the nonlinear wavelet function in the consequent layer. In short, the fuzzy rule is defined as follows:

$$R_i: \textbf{IF } X \textit{ is } \tilde{R}_i \textbf{ Then } \tilde{y}_i^{\,o} = \phi_i(z_i)\tilde{\Omega}_{i,o}, \tilde{\Omega}_{i,o} = [\underline{\Omega}_{i,o}, \overline{\Omega}_{i,o}] \quad (1)$$

where $\tilde{R}_i = [\underline{R_i}, \overline{R_i}]$ stands for a multidimensional kernel with uncertain means written as follows:

$$\tilde{R}_i = \exp(-(X_n - \tilde{C}_i)\Sigma_i^{-1}(X_n - \tilde{C}_i)^T) \;\; \tilde{C}_i = [\underline{C}_{i,1}, \overline{C}_{i,2}] \quad (2)$$

where $\tilde{C}_i = [\underline{C}_i, \overline{C}_i] \in \Re^{1\times u}$ denotes the uncertain centroids of the *i-th* rule and *u* is the number of input dimensions. Because the interval type-2 fuzzy set with uncertain means is employed, the upper centroid is set larger than the lower centroid $\underline{C}_i < \overline{C}_i$. $\Sigma_i^{-1} \in \Re^{u\times u}$ represents a non-diagonal inverse covariance matrix, whose elements pinpoint the interrelation of input variables, and in turn govern the orientation of ellipsoids. The multivariate Gaussian function induces more reliable input space partition, because the non-axis parallel ellipsoidal cluster is capable of covering arbitrary contours of data clouds [3]. This merit reduces the demand of fuzzy rules in the training process. On the other hand, the multivariate Gaussian rule possesses the scale-invariant trait and sustains the interrelation among input attributes, which vanishes under the conventional fuzzy rule using the product t-norm operator.

The multivariate Gaussian rule does not possess the fuzzy set representation. This fact affects to the rule transparency, because the fuzzy rule does not exhibit the atomic clause of human-like linguistic rule. In realm of the interval type-2 fuzzy system, the fuzzy set also plays crucial role in forming the Footprint of Uncertainty (FoU). Hence, a transformation strategy must be carried out to form the fuzzy set representation of the multivariate Gaussian rule. Since the centroid of the

multivariate Gaussian function has the same expression in the fuzzy set level, our only focus is to solicit the radii of the Gaussian function in the main axes. To this end, the second method in [3] is generalised for the interval type-2 fuzzy rule in the eT2RFNN, because it offers an instantaneous mechanism, although it is rather inaccurate in dealing with the ellipsoidal cluster rotated at 45 degrees. In a nutshell, the radius of the non-axis ellipsoidal cluster is defined using the average cardinality principle as follows:

$$\sigma_i = \frac{(\underline{r}_i + \bar{r}_i)}{2\sqrt{\Sigma_{ii}}} \tag{3}$$

where $\Sigma_{i,i}$ denotes the diagonal element of the covariance matrix and $\underline{r}_i, \bar{r}_i$ shows the lower and upper Mahalanobis distance $(X_n - \tilde{C}_i)\Sigma_i^{-1}(X_n - \tilde{C}_i), \tilde{C}_i = [\underline{C}_i, \bar{C}_i]$.

$\tilde{y}_i^o = [\underline{y}_i{}^o, \bar{y}_i{}^o] \in \Re^{1\times m}$ is an interval consequent of the *i-th* local sub-model and is defined as $\underline{y}_i^o = \phi_i(X_N)\underline{\Omega}_i^o, \bar{y}_i^o = \phi_i(X_N)\bar{\Omega}_i^o$. $\tilde{\Omega}_i = [\underline{\Omega}_i, \bar{\Omega}_i] \in \Re^{u\times m}$ which denotes the interval weight vector of the *i-th* rule, where $\bar{\Omega}_i = [\bar{w}_1, ..., \bar{w}_u], \underline{\Omega}_i = [\underline{w}_1, ..., \underline{w}_u]$ and *m* stands for the number of output dimension. $\phi_i(z_i)$ stands for the extended input vector, stemming from a nonlinear mapping of the Wavelet function. Note that the standard zero or first TSK rule consequent does not fully explore the local output approximation aptitude. This flaw is overcome by exploiting the nonlinear Wavelet function. The wavelet function puts forward the multi-resolution property, which can capture useful information on various resolution levels. As a result, it substantiates the model's generalisation and reduces the number of fuzzy rules. We make use of the dilated and translated versions of the Mexican function [23] as follows:

$$\phi_i(z_{j,i}) = \prod_{j=1}^{u}(1 - z_{j,i}{}^2)\exp(-\frac{z_{j,i}{}^2}{2}), Z_i = (\frac{D_i - A_i}{B_i}) \; A_i = [a_{1,1\times u}, ..., a_{P,1\times u}] \in \Re^{P\times u}, B_i = [b_{1,1\times u}, ..., b_{P,1\times u}] \in \Re^{P\times u}, D_i = [D_{1,1\times u}, ..., D_{P,1\times u}] \in \Re^{P\times u} \tag{4}$$

where $a_{i,j}, b_{i,j}$ indicates the dilation and translation parameters and $d_{i,j}$ labels the output of the local recurrent layer. *P* denotes the number of rules. The dilation and translation variables are not fixed, but rather adjusted to portray the different local behaviours of the real trend of the approximation curve. We do not rely on the concept of functional link consequent via the trigonometric or Chebyshev function [24], because this strategy imposes a more prohibitive memory demand due

to the higher DoF and is not adaptive due to the absence of the tuning mechanism to adjust its shape.

$$d_{i,j}(n) = x_j(n) + \phi_{i,j}(n-1)\theta_{i,j}(n) \tag{5}$$

where $\theta_{i,j}$ denotes the weight of the self-feedback link, which can be seen as a storage coefficient. The efficacy of the Wavelet function usually comes at cost of the rule interpretability, because an input vector is mapped to the Wavelet domain – frequency domain. This shortcoming is, slightly, alleviated in the eT2RFNN, because it adopts the local learning scheme. The local learning scheme provides some sort of rule transparency in terms of operating region of the rule consequent, where each rule consequent represents a specific area of the output space in the Wavelet domain.

The interval type-2 fuzzy inference scheme can be committed once eliciting the spread of the multivariate Gaussian function in the main axes, because an interval-valued membership can be produced by computing the upper and lower membership degrees as follows:

$$\tilde{\mu}_{i,j} = \exp(-(\frac{x_j - \tilde{c}_{i,j}}{\sigma_{i.j}})^2), \tilde{c}_i = [\underline{c}_{j,1}{}^i, \bar{c}_{j,2}{}^i] \tag{6}$$

$$\bar{\mu}_{i,j} = \begin{cases} N(c_{j,1}{}^i, \sigma_{i,j}; x_j) & x_j < c_{j,1}{}^i \\ 1 & c_{j,1}{}^i \le x_j c_{j,2}{}^i \\ (c_{j,2}{}^i, \sigma_{i,j}; x_j) & x_j > c_{j,2}{}^i \end{cases}, \tag{7}$$

$$\underline{\mu}_{i,j} = \begin{cases} N(c_{j,2}{}^i, \sigma_{i,j}; x_j) & x_j \le \frac{(c_{j,1}{}^i + c_{j,2}{}^i)}{2} \\ N(c_{j,1}{}^i, \sigma_{i,j}; x_j) & x_j > \frac{(c_{j,1}{}^i + c_{j,2}{}^i)}{2} \end{cases} \tag{8}$$

Henceforth, the interval firing strength $\tilde{R}_i = [\underline{R_i}, \overline{R_i}]$, describing the matching factor or the completeness of fuzzy rule, is obtained by applying the t-norm operator as follows:

$$\underline{R_i} = \prod_{j=1}^{u} \underline{\mu_{i,j}}, \overline{R_i} = \prod_{j=1}^{u} \overline{\mu_{i,j}} \tag{9}$$

The rule firing strength is fed to the temporal firing strength layer, which is equipped with an internal feedback loop. The temporal firing strength layer results in the temporal firing strength $\tilde{\psi}_i^o = [\underline{\psi}_i^o, \overline{\psi}_i^o], i = 1,...,P, o = 1,...,m$, which is not only influenced by the current spatial firing strength but also caused by the past temporal firing strength as follows:

$$\overline{\psi}_i^o(n) = \lambda_i^o \overline{R}_i(n) + (1-\lambda_i^o)\overline{\psi}_i^o(n-1) \ , \underline{\psi}_i^o(n) = \lambda_i^o \underline{R}_i(n) + (1-\lambda_i^o)\underline{\psi}_i^o(n-1) \tag{10}$$

where $\lambda_i^o \in [0,1]$ denotes the weight of the recurrent layer. It is observed that the temporal firing strength is unique for the *i-th* rule of the *o-th* class to foster the temporal learning property of the network architecture. Henceforth, the type reduction mechanism is undertaken to transform the interval-valued set to the crisp variable, called the type-reduced set. The q design coefficients $[q_l, q_r]$ are utilised to perform the type reduction mechanism. The underlying rationale in using the q design factor rather than the KM method is that it offers computationally lighter burden and more flexible characteristics, where it can be directly embedded in the inference process. The type-reduced set, leading to the final crisp output, is written as follows.

$$y_{l,o} = \frac{\sum_{i=1}^{P} \underline{\psi}_i \underline{y}_{i,o} q_l^{\,o}}{\sum_{k=1}^{P} \underline{\psi}_k} + \frac{\sum_{i=1}^{P} \overline{\psi}_i \underline{y}_{i,o} (1 - q_l^{\,o})}{\sum_{k=1}^{P} \underline{\psi}_k}, \; y_{r,o} = \frac{\sum_{i=1}^{P} \overline{\psi}_i \overline{y}_{i,o} (1 - q_r^{\,o})}{\sum_{k=1}^{P} \underline{\psi}_k} + \frac{\sum_{i=1}^{P} \underline{\psi}_i \overline{y}_{i,o} q_r^{\,o}}{\sum_{i=1}^{P} \overline{\psi}_k}, \; y_o = \frac{1}{2}(y_{l,o} + y_{r,o}) \quad (11)$$

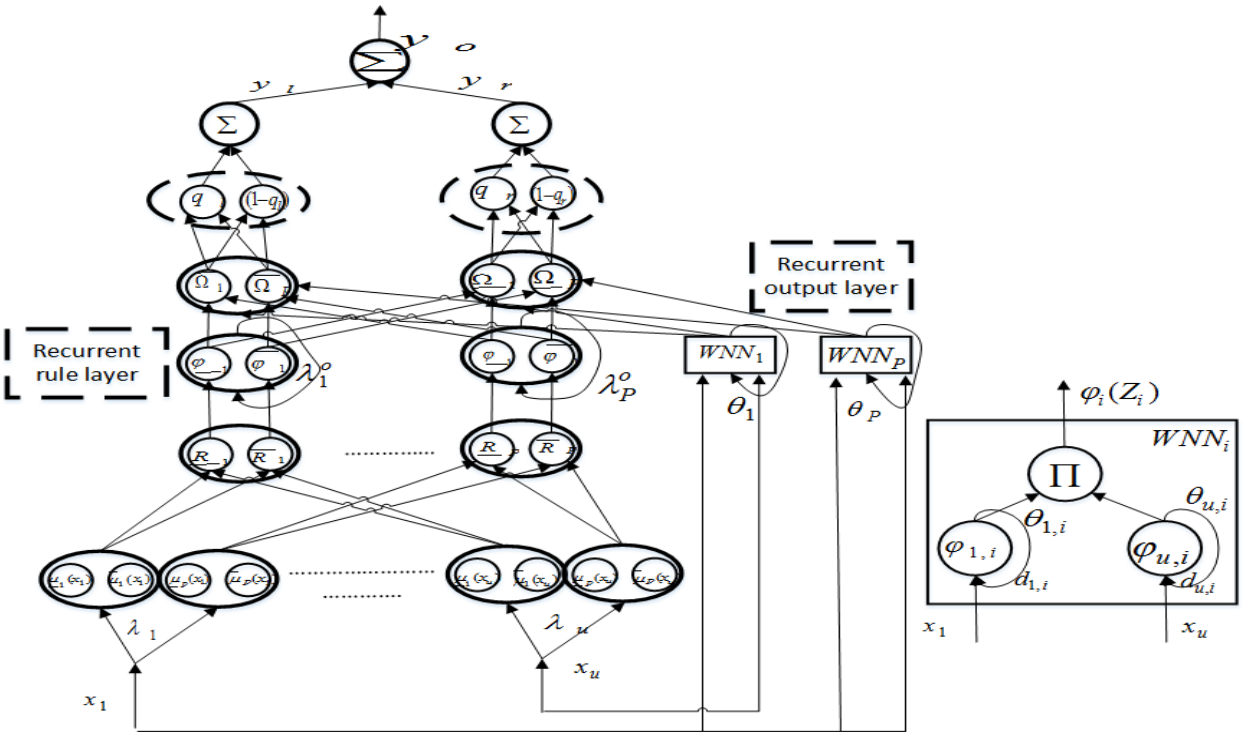

Fig. 1 Network Architecture of eT2RFNN

where $[y_l, y_r]$ stands for the type-reduced set and y_o denotes the final crisp output. The design coefficients $[q_l, q_r]$ are adjusted using the T2ZEDM method to govern the proportion of the upper and lower rules adaptively. The universal approximation characteristic of the eT2RFNN is discussed here.

Remark: the normalisation term of eT2RFNN is amended from those of [9], because the normalisation formula $\overline{A}\Big/\sum_{k=1}^{P}\overline{A}$ in [9] may lead to invalid interval. Suppose that $\tilde{R}_i=[\underline{R}_i,\overline{R}_i]=[0.65,0.75;0.58,0.62;0.34,0.44]$. If we apply $\overline{A}\Big/\sum_{k=1}^{P}\overline{A}$, we arrive at invalid intervals $[0.414,0.414;0.369,0.343;0.22,0.24]$. Our modification $\overline{A}\Big/\sum_{k=1}^{P}\underline{A}$ produces $[0.36,0.48;0.32,0.39;0.19,0.28]$. This strategy is inspired by the interval analysis in [25].

Theorem 1: Universal approximation theorem – our network architecture can predict any real continuous function T in a compact set $U\subset\Re^N$. For any given $\varepsilon>0$, there exists a model, satisfying $\sup_{x\in U}\|f(x)-T(x)\|<\varepsilon$, where $\|\bullet\|$ can be any norm.

This theorem implies that for any given target function over any compact time interval $n\in[n_o,N]$, the eT2RFNN can output uniformly reliable prediction with error smaller than any given value of ε.This theorem can be easily proven using the Stone-Weierstrass theorem as done in the literature [44]. Nevertheless, the Stone-Weierstrass theorem does not yet answer the inter-temporal dependencies of the recurrent network architecture, because it is mainly based on a static input/output mapping. Universal approximation capability of the recurrent network architecture with a specific case of the eT2RFNN is deliberated in Appendix.

IV. Rule Base Management of the eT2RFNN

This section details the learning policy of the eT2RFNN. The pseudo-code of the eT2RFNN is provided in the Algorithm 1.

A) The Rule Growing Module: The eT2RFNN is capable of automatically evolving the fuzzy rules utilising the T2DQ method. The T2DQ method performs knowledge exploratory mechanism by incrementally computing the density of all training data seen thus far and in turn recruits samples offering high generalisation potential and summarisation power. This method relies on the density-based approach, initiated in the subtractive clustering approach [56] and enhanced in the mountain clustering method [57]. Important contribution has been conveyed by Angelov et al in [1], where it transforms the density-based clustering approach for an online rule identification

strategy, the main component of the EFS to cope with the concept drift. The T2DQ method here modifies the density-based approach for the type-2 fuzzy system. It is formulated as follows:

$$FS_{P+1} = \frac{1}{2}\left(\sqrt{\frac{U_N}{U_N(1+b_N) - 2h_N + g_N}} + \sqrt{\frac{U_N}{U_N(1+b_N^{\,1}) - 2h_N^{\,1} + g_N}}\right) \tag{16}$$

$$U_N = U_{N-1} + FS_{N-1},\ b_N = \sum_{j=1}^{u+m} (\bar{c}_{j,P+1})^2,\ b_N^{\,1} = \sum_{j=1}^{u+m} (\underline{c}_{j,P+1})^2,\ h_N = \sum_{j=1}^{u+m} c_{j,P+1} z_{N-1,j},\ h_n^{\,1} = \sum_{j=1}^{u+m} c_{j,P+1} z_{N,j}^{\,1},\ z_{N,j} = z_{N-1,j} + x_{j,N-1} U_{N-1,j}$$

$$g_N, j = g_{N-1,j} + \sum_{j=1}^{u+m} (x_{j,N-1})^2 U_{N-1,j}$$

Algorithm 1: Learning policy of eT2RFNN

Define: Training Data $(X_n, T_n) = (x_1,..,x_u, t_1,..,t_m)$

Predefined Parameters $\omega = 10^5, s = 0.05, \varpi = 10^{-15}$

/*Phase A: Rule Growing and Recall Scenario/*
For i=1 to P do
Compute the T2DQ method (16)
End For
For i=1 to P* do
compute the T2DQ method for P* rules()
End For
Determine the winning rule $win = \arg\max_{i=1,...,P} \hat{P}(R_i|X)$
IF (17) Then
If (27)
Recall the previously pruned fuzzy rule (28)
Else
Add a new rule (18), (19)
End If
Else
Adapt the winning rule (22)-(24)
End If
/*Phase B: Rule Pruning Mechanism/*
For i=1 to P do
Compute RMI (24)
IF (26) Then
Prune *i-th* hidden node
End IF
End For
/*Phase C: Rule Merging Mechanism/*
For i=1 to P do
For j=1 to u do
Compute the shape-based and proximity-based similarity measures(31),(32)
Quantify the vector similarity measure (29)
End For
IF (30),(34) Then
Coalesce the fuzzy rules (35)-(38)
End IF
End For
End IF
/*Phase D: Feature Selection Mechanism/*
For j=1 to u Do
For o=1 to m Do
Compute the symmetrical uncertainty
IF $SU(x_j, t_o) < \gamma$ ***Then***
Prune the *j-th* feature
Else IF
Append *j-th* as relevant variables $x_{re}, N_{re} = N_{re} + 1$
End IF
End For
End For
For j=1 to N_{re}
For re=1 to N_{re}, $j \neq re$
Analyze the redundancy of input attributes $SU(x_j, x_{re})$
IF $SU(x_j, T) < SU(x_j, x_{re})$ ***Then***
Prune *re-th* input variable
End IF
End For
End For
/*Phase E: Parameter Learning Scenario /*
For i=1 to P do
Adjust the fuzzy rule consequents (39)-(42)
For j=1 to u do
Adjust the dilation and translation parameters (45),(46)
Adapt the input recurrent weights (47)
End For
For o=1 to m do
Fine-tune the design factors (43),(44)
Adjust the consequent recurrent weight (48)
End For
End For

where $x_{j,N-1}$ is a datum in the *j-th* input attribute received at *N-1* training observation and $\bar{c}_{j,P+1}, \underline{c}_{j,P+1}$ stands for the centroid of hypothetical rule (P+1st rule), assigned as $\bar{c}_{j,P+1} = x_j^{\,N} + \Delta x$, $\underline{c}_{j,P+1} = x_j^{\,N} - \Delta x$. Δx is an uncertainty factor, determining the footprint of uncertainty of the rule, and is fixed as 0.1 for all our simulations here. $x_j^{\,N}$ is the latest incoming datum in the *j-th* coordinate. We apply the average cardinality principle in (16) which supposes the upper and lower rules to contribute

equally *q=0.5*. All recursive parameters are initialised as zero. We assign the weighting factor U_N to diminish the outlier's leverage as put into perspective in [27]. A training sample is appended as a new rule if it complies with the following criteria.

$$FS_{P+1} > \max_{i=1,\ldots,P}(FS_i) \; or \; FS_{P+1} < \min_{i=1,\ldots,P}(FS_i) \tag{17}$$

where FS_i can be elicited by replacing $\bar{c}_{j,P+1}, \underline{c}_{j,P+1}$ in (16) with $\bar{c}_{j,i}, \underline{c}_{j,i}$. The condition $FS_{P+1} > \max_{i,=1,\ldots,P}(FS_i)$ is deemed to be noteworthy to enhance the summarisation power of the rule base, because it captures a highly dense region. On the other hand, $FS_{P+1} < \min_{i,=1,\ldots,P}(FS_i)$ plays a crucial role to discover the shift of the system dynamic, because it pinpoints a remote region uncharted by the influence zone of the existing rules. Although $FS_{P+1} < \min_{i,=1,\ldots,P}(FS_i)$ opens the likelihood for outliers to be embedded as fuzzy rules, the eT2RFNN is equipped with a rule pruning scenario to overcome the problem. It is worth-mentioning that the T2DQ method can be distinguished from the potential method [1] in three facets: 1) the T2DQ method is equipped by the weighting factor, which helps to reduce the impact of outliers. As found in [27], outliers cause a large pair-wise distance, affecting density estimation of next training samples; 2) the T2DQ method defines the type-2 version of the density-based method, which is fit to the working principle of the type-2 fuzzy system; 3) the T2DQ method is built upon a different kernel function, where the inverse multi-quadratic function is used as an alternate of the Cauchy kernel.

Note that the rule growing scenario functions similarly with a drift detection strategy in the realm of machine learning, because it autonomously partitions the input space with respect to true data distribution. In comparison with other knowledge exploratory modules in the IT2EFS [7],[10], the T2DQ method is more robust against outliers. It enumerates the distance between a new datum and other data seen thus far, whereas most IT2EFSs [7],[10] rely on the rule firing strength to generate fuzzy rules. In addition, the T2DQ method is threshold-free – no problem-dependent threshold needs to be predefined by the user. The parameters of a new fuzzy rule are stipulated:

$$\tilde{C}_{P+1} = X_N \pm \Delta X, diag(\Sigma_{P+1}) = \frac{\max((C_i - C_{i-1}), (C_i - C_{i+1}))}{\sqrt{\frac{1}{\ln(\varepsilon = 0.5)}}} \tag{18}$$

where $\varepsilon = 0.5$ is a completeness constant, which determines a completeness degree a new rule should have. This issue pertains to the minimum firing strength of a rule $(\overline{R}_i + \underline{R}_i)/2 \geq \varepsilon$. $\varepsilon = 0.5$ is selected, because as investigated in [28], the degree of confidence should not be less than 0.5. We initialise the centre of a new fuzzy rule as that of the hypothetical rule $\overline{C}_{P+1} = \overline{C}_{P+1} + \Delta X, \underline{C}_{P+1} = \underline{C}_{P+1} - \Delta X$ and ΔX is fixed as 0.1. The ε completeness criterion is attained using the setting of the covariance matrix as verified in [28]. The new local sub-model and output covariance matrix are assigned:

$$\widetilde{\Omega}_{P+1} = \widetilde{\Omega}_{win} \ \widetilde{\gamma}_{P+1} = \omega I \ , \widetilde{\Omega}_{P+1} = [\underline{\Omega}_{P+1}, \overline{\Omega}_{P+1}], \widetilde{\gamma}_{P+1} = [\underline{\gamma}_{P+1}, \overline{\gamma}_{P+1}] \tag{19}$$

where ω denotes a large positive constant and is fixed as $\omega = 10^5$. $\widetilde{\gamma}_{P+1} = [\underline{\gamma}_{P+1}, \overline{\gamma}_{P+1}] \in \Re^{u \times u}$ is the new output covariance matrix. Setting a new output covariance matrix with (19) is desired, because it is capable of emulating a real solution as attained by the batched learning scheme instantaneously [29]. The new local sub-system is akin to the output covariance matrix of the winning rule because the winning rule is adjacent to the new rule in the context of local learning, thereby potentially being pertinent to the new rule.

The winning rule is selected by the Bayesian concept, where it focuses on a fuzzy rule having a maximum posterior probability $win = \arg\max_{i=1,\ldots,P} \hat{P}(\widetilde{R}_i | X)$. The underlying reason to choose the Bayesian concept in lieu of the omnipresent compatibility measure is because it extracts the winning rule in the probabilistic fashion. When there are several candidates lying on par to a newly received datum, the prior probability will endorse the most populated rule covering more samples. In a nutshell, the posterior probability, the prior probability, and the likelihood function are respectively expressed as follows:

$$\hat{P}(R_k|X) = \frac{1}{2}\left(\frac{\hat{p}(X|\overline{R}_k)\hat{P}(R_k)}{\sum_{i=1}^{P}\hat{p}(X|\overline{R}_i)\hat{P}(R_i)} + \frac{\hat{p}(X|\underline{R}_k)\hat{P}(R_k)}{\sum_{i=1}^{P}\hat{p}(X|\underline{R}_i)\hat{P}(R_i)}\right), \ \hat{P}(R_i) = \frac{\log(N_i+1)}{\sum_{i=1}^{P}\log(N_i+1)}, \ \hat{P}(X|\widetilde{R}_i) = \frac{1}{(2\pi)^{1/2}\widetilde{V}_i^{1/2}}\exp(-(X-\widetilde{C}_i)\Sigma_i^{-1}(X-\widetilde{C}_i)^T) \tag{20}$$

where N_i labels the number of supports of the *i-th* cluster. The expression of the prior probability $\hat{P}(R_i)$ is softened from its original formula with the use of the *log* operation. This technique is to allow a new cluster to compete with older clusters in the winning rule selection phase [26].

It may happen in the training process that a data point violates the rule growing condition (17). This condition triggers the rule premise adaptation as follows:

$$\tilde{C}_{win}^{N} = \frac{N_{win}^{N-1}}{N_{win}^{N-1}+1}\tilde{C}_{win}^{N-1} + \frac{(X_N - \tilde{C}_{win}^{N-1})}{N_{win}^{N-1}+1}, \tilde{C}_{win} = [\underline{C}_{win}, \overline{C}_{win}] \tag{21}$$

$$\Sigma_{win}(N)^{-1} = \frac{\Sigma_{win}(N-1)^{-1}}{1-\alpha} + \frac{\alpha}{1-\alpha}\frac{(\Sigma_{win}(N-1)^{-1}(X_N - \hat{C}_{win}^{N-1}))(\Sigma_{win}(N-1)^{-1}(X_N - \hat{C}_{win}^{N-1}))^T}{1+\alpha(X_N - \hat{C}_{win}^{N-1})\Sigma_{win}(old)^{-1}(X_N - \hat{C}_{win}^{N-1})^T} \tag{22}$$

$$N_{win}^{N} = N_{win}^{N-1} + 1 \tag{23}$$

where $\alpha = 1/(N_{win}^{N-1}+1)$ and $\hat{C}_{win} = (\underline{C}_{win} + \overline{C}_{win})/2$. The adaptation scenario in (22) allows adjusting the inverse covariance matrix directly and overrides the requirement of re-inversion [49]. This mechanism leads to a more stable and faster adaptation. We exploit the midpoint of the interval valued Centre or implement the so-called average cardinality principle to adjust the inverse covariance matrix.

B) Rule Pruning Mechanism and Rule Recall Scenario: A novel rule pruning scenario, namely the T2RMI method, is proposed in this paper, where the key idea is to check the correlation between the fuzzy rule and the target concept. Note that the underlying difference between the T2RMI method and the RMI method in [18] can be seen in its incremental working framework. Furthermore, the original RMI method has been only applied to the T1EFS. The relationship between the two variables can be analysed using either a linear or nonlinear measure. The linear measure is, however, inaccurate, because the interaction between two variables is nonlinear in nature [31]. To this end, we exploit the symmetric uncertainty method, which characterises three appealing traits: simplicity, low bias for multi-valued features, and insensitivity to the order of two variables [32]. The T2RMI method is formulated using the average cardinality principle:

$$RMI(\tilde{R}_i, Y) = \frac{I(\underline{R_i}, Y)}{H(\underline{R_i}) + H(Y)} + \frac{I(\overline{R_i}, Y)}{H(\overline{R_i}) + H(Y)} \tag{24}$$

where $I(\overline{R}_i, Y) = H(\overline{R}_i) + H(Y) - H(\overline{R}_i, Y)$ is the information gain or the mutual information. $H(\overline{R}_i)$ is the entropy of $\overline{R}_i$ and $H(\overline{R}_i, Y)$ is the joint entropy of $\overline{R}_i, Y$. (24) revolves around [0,1], where zero signifies that the two variables are uncorrelated. It is worth noting that the term symmetric here refers to the information gain $I(\overline{R}_i, Y)$, which is a symmetric measure. Changing the order of the

two variables will not affect the information gain formula $I(\bar{R}_i, Y) = I(Y, \bar{R}_i)$. We take this term from the popular definition in [22],[58]. There are several avenues to elicit the entropy. The discretisation method and the Parzen window estimation are among the most renowned methods. Both approaches however adopt a batched learning concept, which is computationally expensive. We use the notion of differential entropy, which assumes uniformly distributed training data [33]:

$$H(\bar{R}_i) = \frac{1}{2}(1 + \log(2\pi \, \mathrm{var}(\bar{R}_i))), \; I(\bar{R}_i, Y) = -\frac{1}{2}\log(1 - \rho^2_{\bar{R}_i Y}), \; \rho_{\bar{R}_i Y} = \frac{\mathrm{cov}(\bar{R}_i, Y)}{\sqrt{\mathrm{var}(\bar{R}_i)\,\mathrm{var}(Y)}} \tag{25}$$

where $\mathrm{var}(\bar{R}_i), \mathrm{cov}(\bar{R}_i, Y), \rho_{\bar{R}_i Y}$ are respectively the variance of $\bar{R}_i$, covariance of $\bar{R}_i, Y$, and Pearson's correlation coefficient. It is worth mentioning that the variance of $\bar{R}_i$ and covariance of $\bar{R}_i, Y$ can be enumerated recursively. The fuzzy rule is pruned given that the following condition is satisfied.

$$RMI_i < mean(RMI_i) - 2std(RMI_i), \; mean(RMI_i) = \frac{\sum_{n=1}^{N} RMI_{i,n}}{N}, \; std(RMI_{i,n}) = \sqrt{\frac{\sum_{n=1}^{N}(RMI_{i,n} - mean(RMI_i))^2}{N-1}} \tag{26}$$

The mean and standard deviation are computed in respect to the RMI during its lifespan and recursively with ease. Condition (26) can also be interpreted as a measure of significant downtrend in the correlation of the *i-th* rule and the target feature. The T2RMI method focusses on the relevance of a fuzzy rule for the target concept. This feature is important to capture the drift of the target concept, the most common type of concept drift. Once pruned, the fuzzy rule is not permanently forgotten, and is able to be reactivated when it is fit to the current data concept $P^* = P^* + 1$. P^* denotes the number of fuzzy rules pruned by (24). This scenario is an important strategy to respond recurring concept drift, because evolving a completely new fuzzy rule to deal with the previously learned concept undermines the logic of truly adaptive online systems and catastrophically omits the learning history. In a nutshell, the rule recall criterion is committed when the following criterion is met.

$$\max_{i^*=1,..,P^*}(FS_{i^*}) > \max_{i=1,..,P+1}(FS_i) \tag{27}$$

The pruned fuzzy rule is maintained in memory to solely quantify (16) and is ruled out in any other learning scenario. Accordingly, computational load is still relieved. The pruned fuzzy rule is reactivated by setting the pruned rule as a new rule as follows:

$$C_{P+1} = C_{i*}, \tilde{\tilde{\Sigma}}_{P+1}^{-1} = \tilde{\tilde{\Sigma}}_{i*}^{-1}, \tilde{\tilde{\gamma}}_{P+1} = \tilde{\tilde{\gamma}}_{i*}, \tilde{\tilde{\Omega}}_{P+1} = \tilde{\tilde{\Omega}}_{i*} \quad (28)$$

C) Rule Merging Mechanism: Several rule merging methodologies have been designed in the context of interval type-2 EFS [34],[35], where they are constructed by a shape-based or distance-based similarity measure without synergising the two approaches. These methods cannot detect the problem of a non-homogenous cluster, where merging different orientation clusters causes a blow-up effect. This phenomenon is more apparent in a non-axis parallel ellipsoidal cluster, because it enables the cluster to arbitrarily rotate in any direction. To remedy these bottlenecks, the eT2RFNN is equipped with a multi-faceted strategy: a vector similarity measure and a blow-up check. The vector similarity measure estimates the similarity between the two clusters based on their shape and proximity, while non-homogenous clusters are solved by the blow-up check.

The vector similarity measure enumerates the similarity between two fuzzy rules more accurately with the use of their distance and shape in one joint formula as follows:

$$s_{v,j}(win,i) = s_{1,j}(win,i) \times s_{2,j}(win,i) \quad (29)$$

where $s_{1,j}(win,i) \in [0,1]$ is the shape-based similarity measure between the winning rule and the *i-th* rule in the *j-th* coordinate and $s_{2,j}(win,i) \in [0,1]$ is the distance-based similarity measure between the winning rule and the *i-th* rule in the *j-th* axis. We merely investigate the similarity between the winning rule and other rules to expedite the model update, because the winning rule is the sole rule to receive the rule premise adaptation, the main reason for the rule overlapping. The similarity measure in the rule level is elicited by combining the fuzzy set similarity in each input dimension with the use of the *t-norm* operator as follows:

$$S_v \geq \rho_3, S_v = \min_{j=1,..,u}(s_{v,j}) \quad (30)$$

where $\rho_3 \in [0,1]$ stands for a predefined threshold set as $\rho_3 = 0.5$ in all our simulations in this paper. It is worth noting that this parameter is not problem-specific and has been confirmed in [17].

Since the vector similarity method treats shape and distance as two independent issues. The alignment procedure is carried out in the shape-based similarity measure, thereby being able to compare the contour of two fuzzy sets more precisely. That is, the centroid of the winning rule $c_{win,j}$ and the *i-th* rule $c_{i,j}$ is aligned as $c_{win,j} = c_{i,j}$. The shape-based similarity between the two rules

is quantified with an extended Jaccard similarity measure, making use of the average cardinality principle as follows:

$$s_{1,j}(win,i)=\frac{M(\underline{\mu_{win,j}}\cap\underline{\mu_{i,j}})+M(\overline{\mu_{win,j}}\cap\overline{\mu_{i,j}})}{M(\underline{\mu_{win,j}}\cup\underline{\mu_{i,j}})+M(\overline{\mu_{win,j}}\cup\overline{\mu_{i,j}})} \tag{31}$$

where $\cup$and $\cap$respectively stand for the union and intersection of the two fuzzy sets $\tilde{\mu}_{win}, \tilde{\mu}_i$. The union of the two fuzzy sets can be derived as $M(\underline{\mu_{win,j}}\cup\underline{\mu_{i,j}})=M(\underline{\mu_{win,j}})+M(\underline{\mu_{i,j}})-M(\underline{\mu_{win,j}}\cap\underline{\mu_{i,j}})$ and $M(.)$ denotes the area of the fuzzy set. Because the Gaussian function has a highly nonlinear contour, its size is approximated using the triangular function $M(\underline{\mu_{win,j}})=\int_{-\infty}^{\infty}\exp(-\frac{(x-c)^2}{2\sigma^2})dx=\sigma_{win,j}{}^1\sqrt{2\pi}$. The union of two fuzzy sets is obtained as follows:

$$M(\underline{\mu_{win,j}}\cap\underline{\mu_{i,j}})=\frac{h^2}{2}+\frac{h^2((\underline{\sigma}_{win,j}-\underline{\sigma}_{i,j}))}{2(\underline{\sigma}_{i,j}-\underline{\sigma}_{win,j})}-\frac{h^2(\underline{\sigma}_{win,j}+\underline{\sigma}_{i,j})}{2(\underline{\sigma}_{win,j}-\underline{\sigma}_{i,j})} \tag{32}$$

where $h=\max[0,x]$. We can execute (32) by performing similar mathematical operations for the upper fuzzy set $M(\overline{\mu_{win,j}}\cap\overline{\mu_{i,j}})$ and $M(\overline{\mu_{win,j}}\cup\overline{\mu_{i,j}})$.On the other hand, the distance-based similarity measure is committed using the extended Kernel-based metric, where the type-1 version of the kernel-based metric was developed in [3]. This method can be applied for the interval type-2 fuzzy system by virtue of the average cardinality principle as follows:

$$S_{2,j}(win,i)=\frac{\exp(-A)+\exp(-B)}{2} \tag{33}$$

$$A=\left|c_{win,j}{}^1-c_{i,j}{}^1\right|-\left|\overline{\sigma}_{win,j}-\overline{\sigma}i,j\right| \quad B=\left|c_{win,j}{}^2-c_{i,j}{}^2\right|-\left|\underline{\sigma}_{win,j}-\underline{\sigma}_{i,j}\right|$$

The extended kernel-based metric features the following properties.

$$S_{2,j}(A,B)=1\Leftrightarrow|C_A-C_B|+|\sigma_A-\sigma_B|=0\Leftrightarrow C_A=C_B\wedge\sigma_B=\sigma_B,\ S_{2,j}(A,B)<\varepsilon\Leftrightarrow|C_A-C_B|>\delta\vee|\sigma_A-\sigma_B|>\delta$$

After eliciting both the shape- and distance-based similarities in (31), (33), we finally arrive at the vector similarity measure formula (30).

Merging two different-orientation clusters should be avoided, because it harms the local data representation which affects generalisation potential. This step incurs an over-sized cluster, which leads to a cluster delamination case, in which some local data clouds are contained by a too-large cluster. In light of this issue, the blow-up check is undertaken by examining the volume of the

merged cluster in contrast with their independent volumes. The rule merging scenario proceeds if the following condition is satisfied.

$$\bar{V}_{merged} + \underline{V}_{merged} \leq u((\bar{V}_{win} + \bar{V}_i) + (\underline{V}_{win} + \underline{V}_i)) \tag{34}$$

Equation (35) illustrates that the blow-up situation is unlikely to occur because the volume of the merged cluster does not exceed their independent volumes. We involve the term *u* in (34) to hedge the curse of dimensionality.

If a training observation complies with (30), (34), the two fuzzy rules are merged as follows:

$$\tilde{C}_{merged}^{new} = \frac{\tilde{C}_{win}^{old} N_{win}^{old} + \tilde{C}_i^{old} N_i^{old}}{N_{win}^{old} + N_i^{old}} \;, \tilde{C}_i = [\underline{C}_i, \bar{C}_i] \tag{35}$$

$$\Sigma^{-1}{}_{merged}^{new} = \frac{\Sigma^{-1}{}_{win}^{old} N_{win}^{old} + \Sigma^{-1}{}_i^{old} N_i^{old}}{N_{win}^{old} + N_i^{old}} \tag{36}$$

$$N_{merged}^{new} = N_{win}^{old} + N_i^{old} \tag{37}$$

$$\tilde{\Omega}_{merged}^{new} = \frac{\tilde{\Omega}_{win}^{old} N_{win}^{old} + \tilde{\Omega}_i^{old} N_i^{old}}{N_{win}^{old} + N_i^{old}} \;, \tilde{\Omega}_i = [\underline{\Omega}_i, \bar{\Omega}_i] \tag{38}$$

The rule merging process refers to the weighted average strategy, which weights the cluster as its accumulated supports. This strategy reflects the fact that a cluster with more populations should be more influential to the final shape and orientation of the merged cluster, because the merged cluster should represent the underlying data distribution and foster cluster's populations.

D) Online Feature Selection Mechanism: high dimensional data poses a lot of challenges for most real-world applications and lead to the curse of dimensionality. Therefore, online feature selection, which is capable of discarding spurious input features in the sample-wise manner during the training process, is highly demanded in the EFS to reduce both problem and network complexities. Online feature selection has been incorporated in the EFS encompassing the input pruning scenario [20] or the input weighting concept [21]. Existing methods merely focus on the issue of relevance, while overlooking redundancy problems. It may happen that an input variable is inconsequential, because it shares a strong similarity with other input attributes. Such input attributes can be pruned without significant loss of accuracy.

A new online feature selection approach, namely SMBC, is mounted in the eT2RFNN and presents a sequential version of MBC. The SMBC is inspired by the Markov Blanket Criterion

[22], which synergises relevance and redundancy tests in determining the significance of input features. The SMBC corrects the instability drawback of conventional input pruning scenarios [20], because it assures that once observing the Markov Blanket Criterion in the previous episodes an input feature is no longer required for model updates in the future. Note that although the input weighting approach [21] does not suffer from the instability problem, it still retains the superfluous feature in the training process.

According to the SMBC, an input attribute can be classified into 4 categories according to its contribution: irrelevant, weakly relevant, weakly relevant but non-redundant, and strongly relevant. An optimal input feature subset comprises strongly relevant features and weakly relevant but non-redundant features. Hence, the underlying goal of the SMBC is to find irrelevant and weakly relevant features in order to be pruned. Two tests, namely C-correlation and F-correlation, are developed to implement the SMBC.

Definition 1(C-correlation) [22]: *The correlation between any feature* x_j *and the class* T *is termed as C-correlation, denoted by* $SU(x_j, T)$. *This strategy is used to probe relevance.*

Definition 2 (F-correlation) [22]: *The correlation between any pair of features* $x_j, x_i \ (i \neq j)$ *is termed F-correlation, labelled by* $SU(x_j, x_i)$. *The approximation of redundancy is carried out by the F-correlation.*

First, the C-correlation is performed to vet the relevance of input attributes to the given problem. The input feature, which happens to be irrelevant or weakly relevant $SU(x_j, T) < \gamma$, can be pruned. γ is a input pruning threshold, set as 0.8 according to the recommendation of [22]. The SMBC continues with the F-correlation, examining the redundancy of input variables. Let x_j be an input feature of interest and let $M_n \subset X \ (x_j \notin M_n)$. M_n is claimed to be a Markov Blanket for x_j if it summarises not only the information of x_j to the target class T, but also almost all of the input features. Therefore, the redundant input features are those of the slightly relevant features having the Markov blanket M_n. Specifically, these features are those of coming through the C-correlation test $SU(x_j, T) \geq \gamma$ but violating $SU(x_j, T) \geq SU(x_j, x_i)$. A two-staged process is committed in the

ET2RFNN to speed up the selection process, because the C-correlation is capable of eliminating the irrelevant or weakly relevant features in the first phase, thereby possibly engaging a smaller number of features in the second phase. Both the C-correlation and the F-correlation can be realised using the symmetrical uncertainty method as with the T2RMI methods. We do not detail their mathematical formulas due to space constraints.

E) Adaptation of Network Parameters: The adaptation of the eT2RFNN rule base parameters comprises two parts: Fuzzily Weighted Generalised Recursive Least Square (FWGRLS), Type-2 Zero Error Density Maximisation (T2ZEDM). The FWGRLS method adapts the consequent weight and presents a local learning version of the GRLS method [36]. The peculiar feature of the FWGRLS method in comparison with the standard RLS method lies in the implicit weight decay term, which aims to sustain the weight vector to hover around a small bounded interval. Note that a small and bounded weight vector prevents the model to be unstable and helps to improve the model generalisation [36]. Apart from substantiating the model's generalisation, this technique is able to refine the compactness of the network architecture, because an inactive fuzzy rule possesses a minor weight vector, thereby being detected by the rule pruning method more easily. The FWGRLS method is defined as follows:

$$\widetilde{\psi}(n) = \widetilde{\gamma}_i(n-1)F(n)\left(\frac{\Delta(n)}{\widetilde{R}_i(n)} + F(n)\widetilde{\gamma}_i(n-1)F^T(n)\right)^{-1} \tag{39}$$

$$\widetilde{\gamma}_i(n) = \widetilde{\gamma}_i(n-1) - \widetilde{\psi}(n)F(n)\widetilde{\gamma}_i(n-1) \tag{40}$$

$$\Omega_i(n) = \Omega_i(n-1) - \varpi\Psi_i(n)\nabla\xi(\Omega_i(n-1)) + \Psi(n)(t(n) - y(n)) \tag{41}$$

$$y(n) = x_{en}\Omega_i(n) \text{ and } F(n) = \frac{\partial y(n)}{\partial\Omega(n)} = x_{en} \tag{42}$$

where $\widetilde{R}_i = [\underline{R}_i, \overline{R}_i]$ is the firing strength of the *i-th* rule and $\widetilde{\psi}(n) = [\underline{\psi}(n), \overline{\psi}(n)]$ is the Kalman gain. $\widetilde{\gamma}_i(n) = [\underline{\gamma}_i(n), \overline{\gamma}_i(n)]$, $\Delta(n) \in \Re^{u\times u}$ are respectively the output covariance matrix and the covariance matrix of modelling error. For simplicity, the covariance matrix of modelling error is set as the Hessian matrix as with [36]. $\nabla\kappa(\widetilde{\Omega}_i(n-1))$ stands for the gradient of the weight decay function. Note that we extend the gradient of the weight decay function to the *n-1* time step, when the solution of the gradient is too complex to be obtained. The weight decay function can be assigned as any nonlinear function, which may not be differentiable. The quadratic weight decay function is chosen

here $\kappa(\tilde{\Omega}_i(n-1)) = \frac{1}{2}(\tilde{\Omega}_i(n-1))^2$, because it is capable of diminishing the weight vector to its current values proportionally [36]. Note that the FWGRLS method is different from a prominent FWRLS method [1], because the regularized term, namely the weigth decay term, is incorporated in the cost function of the RLS method. This modification leads to the presence of the weight decay term in (41), which does not exist in the FWRLS method.

The T2ZEDM method adjusts the Wavelet parameters $(a_{i,j}, b_{i,j})$, the design coefficients $(q_{i,o}^{l}, q_{i,o}^{r})$, and the recurrent weights $(\theta_{i,j}, \lambda_{i,j})$. It enhances the standard gradient descent method by modifying its cost function, where the error entropy is employed in lieu of the Mean Square Error (MSE). In principle, this method forces the error entropy to converge at the origin by minimising the distance between the probability distribution of the system's output and the target function. This approach is deemed to be efficient to improve the prediction of high order statistical behaviour. Because the distribution of the error entropy is unknown, the cost function of the T2ZEDM is estimated using the Parzen window estimation method as follows:

$$\hat{f}(0) = \frac{1}{Nh\sqrt{2\pi}} \sum_{n=1}^{N} \exp(-\frac{{e_{n,o}}^2}{2\Gamma^2}) = \frac{1}{Nh\sqrt{2\pi}} \sum_{n=1}^{N} K(\frac{-{e_{n,o}}^2}{2\Gamma^2}) \tag{43}$$

where *N* is the number of training data seen so far and Γ is a smoothing parameter, simply fixed as 1. $e_{n,o}$ is the system error of the *o-th* output in the *n-th* training episode. The optimisation process is carried out using the gradient descent method as follows:

$${q_{l,r}}^{i,o}(N) = {q_{l,r}}^{i,o}(N-1) + \eta_q \frac{\partial \hat{f}(0)}{\partial {q_{l,r}}^{i,o}} = {q_{l,r}}^{i,o}(N-1) - \eta_q \frac{1}{N\sqrt{2\pi}} \sum_{n=1}^{N} K(-\frac{{e_{n,o}}^2}{2}) \frac{\partial E}{\partial {q_{l,r}}^{i,o}} \tag{44}$$

$${a_i}^j(N) = {a_i}^j(N-1) + \eta_a \frac{\partial \hat{f}(0)}{\partial {a_i}^j} = {a_i}^j(N-1) - \eta_a \frac{1}{N\sqrt{2\pi}} \sum_{n=1}^{N} K(-\frac{{e_{n,o}}^2}{2}) \frac{\partial E}{\partial {a_i}^j} \tag{45}$$

$${b_i}^j(N) = {b_i}^j(N-1) + \eta_b \frac{\partial \hat{f}(0)}{\partial {b_i}^j} = {b_i}^j(N-1) - \eta_b \frac{1}{N\sqrt{2\pi}} \sum_{n=1}^{N} K(-\frac{{e_{n,o}}^2}{2}) \frac{\partial E}{\partial {b_i}^j} \tag{46}$$

$${\theta_i}^j(N) = {\theta_i}^j(N-1) + \eta_\theta \frac{\partial \hat{f}(0)}{\partial {\theta_i}^j} = {\theta_i}^j(N-1) - \eta_\theta \frac{1}{N\sqrt{2\pi}} \sum_{n=1}^{N} K(-\frac{{e_{n,o}}^2}{2}) \frac{\partial E}{\partial {\theta_i}^j} \tag{47}$$

$${\lambda_i}^j(N) = {\lambda_i}^j(N-1) + \eta_\lambda \frac{\partial \hat{f}(0)}{\partial {\lambda_i}^j} = {\lambda_i}^j(N-1) - \eta_\lambda \frac{1}{N\sqrt{2\pi}} \sum_{n=1}^{N} K(-\frac{{e_{n,o}}^2}{2}) \frac{\partial E}{\partial {\lambda_i}^j} \tag{48}$$

where $\eta_q, \eta_a, \eta_b, \eta_\theta, \eta_\lambda$ are the adaptive learning rates, determined using the Lyapunov stability criteria to warrant the convergence and E stands for the error function $\frac{(t_n - y_n)^2}{2}$. Because

$\sum_{n=1}^{N}\exp(-\frac{e_{n,o}^2}{2})$ necessitates revisiting preceding training data, we formulate its recursive expression as $\sum_{n=1}^{N}\exp(\frac{-e_n^2}{2}) = A_N = A_{N-1} + \exp(\frac{-e_{N,o}^2}{2})$. The gradient term $\partial E/\partial a_{i,j}, \partial E/\partial b_{i,j}, \partial E/\partial q_{i,r}^{i,o}, \partial E/\partial \theta_{i,j}, \partial E/\partial \lambda_{i,j}$ can be obtained using the chain rule as follows:

$$\frac{\partial E}{\partial q_o^l} = (y_n - t_n)(\frac{\sum_{i=1}^{P}\overline{\psi}_i \underline{y}_{i,o}}{\sum_{k=1}^{P}\overline{\psi}_i} - \frac{\sum_{i=1}^{P}\underline{\psi}_i \underline{y}_{i,o}}{\sum_{k=1}^{P}\underline{\psi}_i}), \frac{\partial E}{\partial q_o^l} = (y_n - t_n)(\frac{\sum_{i=1}^{P}\overline{\psi}_i \overline{y}_{i,o}}{\sum_{k=1}^{P}\overline{\psi}_i} - \frac{\sum_{i=1}^{P}\underline{\psi}_i \overline{y}_{i,o}}{\sum_{k=1}^{P}\underline{\psi}_i}) \tag{49}$$

$$\frac{\partial E}{\partial a_{i,j}} = \frac{\partial E}{\partial y_o}\frac{\partial \phi_i}{\partial z_{i,j}}\frac{\partial z_{i,j}}{\partial a_{i,j}}\left\{(\frac{\partial y_o}{\partial y_l}\frac{\partial y_l}{\partial \underline{y}}\frac{\partial \underline{y}}{\partial \phi_i}) + (\frac{\partial y_o}{\partial y_r}\frac{\partial y_r}{\partial \overline{y}}\frac{\partial \overline{y}}{\partial \phi_i})\right\} = (y_n - t_n)(\frac{-1}{b_{i,j}})(z_{i,j}\exp(\frac{-z_{i,j}^2}{2})(-3 + z_{i,j}^2)) \tag{50}$$
$$\left\{(\frac{\sum_{i=1}^{P}\underline{\psi}_i \underline{\Omega}_i^o (1-q_l^o)}{\sum_{k=1}^{P}\underline{\psi}_k} + \frac{\sum_{i=1}^{P}\overline{\psi}_i \underline{\Omega}_i^o q_i^o}{\sum_{k=1}^{P}\overline{\psi}_k}) + (\frac{\sum_{i=1}^{P}\overline{\psi}_i \overline{\Omega}_i^o (1-q_r^o)}{\sum_{k=1}^{P}\overline{\psi}_k} + \frac{\sum_{i=1}^{P}\underline{\psi}_i \overline{\Omega}_i^o q_r^o}{\sum_{k=1}^{P}\underline{\psi}_k})\right\}$$

$$\frac{\partial E}{\partial b_{i,j}} = \frac{\partial E}{\partial y_o}\frac{\partial \phi_i}{\partial z_{i,j}}\frac{\partial z_{i,j}}{\partial b_{i,j}}\left\{(\frac{\partial y_o}{\partial y_l}\frac{\partial y_l}{\partial \underline{y}}\frac{\partial \underline{y}}{\partial \phi_i}) + (\frac{\partial y_o}{\partial y_r}\frac{\partial y_r}{\partial \overline{y}}\frac{\partial \overline{y}}{\partial \phi_i})\right\} = z_{i,j}(d_{i,j} - b_{i,j})\frac{\partial E}{\partial a_{i,j}} \tag{51}$$

$$\frac{\partial E}{\partial \theta_{i,j}} = \frac{\partial E}{\partial y_o}\frac{\partial \phi_i}{\partial z_{i,j}}\frac{\partial z_{i,j}}{\partial d_{i,j}}\frac{\partial d_{i,j}}{\partial \theta_{i,j}}\left\{(\frac{\partial y_o}{\partial y_l}\frac{\partial y_l}{\partial \underline{y}}\frac{\partial \underline{y}}{\partial \phi_i}) + (\frac{\partial y_o}{\partial y_r}\frac{\partial y_r}{\partial \overline{y}}\frac{\partial \overline{y}}{\partial \phi_i})\right\} = -\phi_{i,j}(n-1)\frac{\partial E}{\partial a_{i,j}} \tag{52}$$

$$\frac{\partial E}{\partial \lambda_{i,j}} = \frac{\partial E}{\partial y}(\frac{\partial y}{\partial \overline{\psi}_i}\frac{\partial \overline{\psi}_i}{\partial \lambda_{i,j}} + \frac{\partial y}{\partial \underline{\psi}_i}\frac{\partial \underline{\psi}_i}{\partial \lambda_{i,j}}) = (y_n - t_n)(((\overline{R}_i - \overline{\psi}_i(n-1))\frac{\phi_i(X_N)\overline{\Omega}_i^o - y_{o,n}}{\sum_{k=1}^{P}\overline{\psi}_k}) + ((\underline{R}_i - \underline{\psi}_i(n-1))\frac{\phi_i(X_N)\underline{\Omega}_i^o - y_{o,n}}{\sum_{k=1}^{P}\underline{\psi}_k})) \tag{53}$$

It is worth stressing that choosing a suitable learning rate plays a crucial role in achieving asymptotic convergence. To this end, the stable range of the learning rate is canvassed with the use of the Lyapunov stability criterion to guarantee convergence. We arrive at $0 < \eta_W < \frac{2N\sqrt{2\pi}}{(P_{Wo,\max})^2 A_N}$. The mathematical proof can be derived following the same steps in [15].

The learning rates are not fixed during the training process, rather are adapted to speed up convergence. Specifically, the learning rates are adjusted using the direction of the entropy cost function (43) $\hat{f}(0)^N$. They augment if the cost function increases $\hat{f}(0)^N > \hat{f}(0)^{N-1}$ and vice versa.

$$\eta_W(N) = \begin{cases}\rho_1\eta_W(N-1), \hat{f}(0)^N \geq \hat{f}(0)^{N-1} \\ \rho_2\eta_W(N-1), \hat{f}(0)^N < \hat{f}(0)^{N-1}\end{cases}, \text{where } 0 < \rho_2 < 1 < \rho_1 \tag{54}$$

where $\rho_5 \in (1,1.5]$, $\rho_4 \in [0.5,1)$ are learning rate factors, which steer the dynamic of the learning rates. Because these factors are not problem-specific, they are simply set as $\rho_1 = 1.1$, $\rho_2 = 0.9$. This setting

is plausible, because the adaptation should be more intense when the error entropy grows to expedite the model's updates.

F) Sensitivity Analysis of Predefined Parameters: the eT2RFNN contains three predefined parameters, $\Delta X, \gamma, \rho_3$, which may hinder the end-user to operate the algorithm. This section aims to ensure that these parameters are not problem-dependent. Furthermore, our numerical results in Section 5 are achieved by setting them at their default values. We exclude ρ_3 from our analysis here, because its effect has been well-studied in [17] and has been shown to be not case-sensitive. ΔX is an uncertainty factor and is commonly used in the interval type-2 fuzzy neural network to determine the region of uncertainty of a new rule [16],[36] .This parameter is set as 0.1, adopting the same setting in the literature [16], [36]. The smaller the value of ΔX the crisper and more specific the new fuzzy set will be created, and vice versa. A too crisp fuzzy set, however, leads a fuzzy set to behave like the type-1 fuzzy set, whereas a too general fuzzy set usually causes loss of representation in a local region. On the other hand, γ is an input pruning threshold, governing the C-correlation of the online feature selection process. The higher the value of this parameter the higher the number of fuzzy rules are discarded during the training process and vice versa. We assign $\gamma = 0.8$ for all our simulations and adopt the same setting of [22].

The sensitivity of the two predefined paramaters is examined using a popular benchmark problem, the Mackey Glass (MG) chaotic time series problem. Our experimental procedure in this paper is akin to the standard configuration in the literature [1]. We varied $\Delta X = [0.05, 0.08, 0.1, 0.2, 0.3]$ to illustrate its effect to the eT2RFNN's learning performance, while we assigned $\gamma = [0.6, 0.7, 0.8, 0.9, 1]$. Note that we kept other parameters fixed at its default setting, when a particular pameter was varied. The learning performance of the eT2RFNN was assessed in three viewpoints: NDEI, fuzzy rule, input attributes. Numerical results are reported in Table 1.

It is clear from Table 1 that different values of $\Delta X, \gamma$ affected little to the eT2RFNN's learning performance. As expected, a small γ resulted in no fuzzy rule to be eliminated during the training process, while a large ΔX deteriorated the accuracy as a result improper input space partion.

G) Complexity Analysis: this section discusses the computational and structural burdens of eT2RFNN, which can be understood from the resultant complexity of learning modules and the network architecture of eT2RFNN. Overall, eT2RFNN entails the computational power of $O(u+u^2m+Pm^2+3Pmu+5m+2P+2Pu+P*mu+P*m^2)$ resulted from the total computational burden of each learning module, and is more importantly independent from the number of training data. The structural complexity of eT2RFNN is determined from the network parameters to be stored in memory. The double recurrent network architecture of eT2RFNN generates $O(2Pu+Pu^2+3Pum+Pu+2Pm)$ network parameters. In conjunction to state-of-the art fuzzy neural networks such as SEIT2FNN [16], SLFRWNN [18], MRIT2NFS [36], eT2Class [17], the eT2RFNN network topology produces a slightly higher number of network parameters. Nonetheless, the number of network parameters is also influenced by the number of evolved fuzzy rules in the training process, where the eT2RFNN is expected to scatter a fewer number of rules. These merits are demonstrated in our numerical studies.

Table 1. Sensitivity of Predefined Parameters

Parameters	*EC*	*I=1*	*I=2*	*I=3*	*I=4*	*I=5*
ΔX	NDEI	0.33	0.33	0.3	0.34	0.35
	R	5	5	3	3	3
	Input	3	3	3	3	3
γ	NDEI	0.32	0.32	0.3	0.3	0.32
	R	3	3	3	3	3
	Input	4	4	3	2	1

CR= Classification rate, R=Rule, RT=Runtime

V. Proof of Concepts

The efficacy of eT2RFNN was experimentally validated using four synthetic and real-world problems, exemplifying various concept drifts, uncertain and noisy characteristics. The proposed algorithm was also benchmarked with prominent machine learning algorithms against five criteria: predictive accuracy, number of fuzzy rules, number of rule base parameters number of input attributes, and execution time. Our simulations were carried out under MATLAB with an Intel (R) core (TM) i7–2600 CPU, 3.4 GHz processor and 8 GB memory.

A. Example 1: Prediction of Nox Emission of Car Engine

A real-world problem, namely prediction of Nox emission of a car engine, is put into perspective to evaluate the viability of the eT2RFNN. This problem features highly noisy and uncertain characteristic, because data were sampled from raw data of a car engine. Furthermore,

this problem also contains a non-stationary characteristic because two important attributes of the engine control, namely rotation speed and torque, were varied in order for different driving behaviours to be represented in the engine data. This study case also offers an appealing property to validate the efficacy of the online feature selection process, because 170 input attributes were extracted to guide the prediction problem. This figure results from 17 different physical variables, recorded in 10 consecutive measurements by hard sensors mounted in the car engine. eT2RFNN is compared with 11 state-of-the art algorithms: PANFIS [3] , GENEFIS [37], simp_eTS [2], simp_eTS+ [62], eTS [1], AnYa [38], eTS+ [20], DENFIS [39], BARTFIS [40], eT2Class [17], DENFIS [39]. We also experimented five learning configurations of eT2RFNN to demonstrate the effectiveness of each learning modules: A) This configuration reveals the learning performance of eT2RFNN without structural complexity reduction method elaborated in Section III.B and C, B) we switch off the online feature selection module outlined in Section III.D, C) the eT2RFNN is implemented using the standard recurrent interval type-2 network architecture [16], C1) this configuration is akin to C but during the training process the same number of rules is evolved to ensure fair comparision with the eT2Class, D) the eT2RFNN is structured in the type-1 network architecture of GENEFIS [37], E) The eT2RFNN utilises only the previous-time instant data *n-1* in the regression model to demonstrate the spatio-temporal property of the eT2RFNN. This problem consists of 826 data points, in which 667 samples are fed in the training process, whereas the remainder is used in the testing samples. The numerical results are reported in Table 1. The predictive quality of benchmarked models is measured using the RMSE.

From Table 2, the eT2RFNN(A)-(E) produced higher accuracy, while retaining lower complexity than other algorithms. The curse of dimensionality is noticeable in this problem, where all algorithms without the feature selection process experienced sluggish execution time. Each learning module contributed substantially to the resultant learning performance of eT2RFNN. It can be viewed from the deterioration of numerical results due to the absence of a particular module. Because of very noisy nature of raw engine data, the learning performance of the eT2RFNN in the type-1 fuzzy system – eT2RFNN(D)- deteriorated substantially. This fact reveals the advantage of

the interval type-2 fuzzy system in processing information uncertainty. Although the eT2RFNN(E) relied on merely the latest time instant data, it still attained comparable numerical results.

Table 2. Prediction of Nox emission

Model	Type	RMSE	Rule	Input	Runtime	Parameters
eT2RFNN	S-2-R	**0.03**	**2**	**1**	0.24	14
eT2RFNN (A)	S-2-R	0.035	4	**1**	0.27	28
eT2RFNN(B)	S-2-R	0.05	2	170	2.13	59840
eT2RFNN (C)	S-2-R	0.04	3	**1**	0.24	18
eT2RFNN(C1)	S-2-R	0.05	**2**	**1**	**0.23**	**12**
eT2RFNN(D)	S-1-R	0.14	5	**1**	0.27	30
eT2RFNN(E)	S-2-R	**0.03**	5	**1**	0.26	35
eT2Class	S-2-F	0.045	2	170	17.98	117304
PANFIS	S-1-F	0.052	5	170	3.37	146205
GENEFIS	S-1-F	0.048	2	2	0.41	18
Simp_eTS+	S-1-F	0.15	15	7	*	330
eTS	S-1-F	0.38	27	170	1098.4	13797
AnYa	S-1-F	0.054	1	43	*	44
eTS+	S-1-F	0.14	4	4	*	52
ANFIS	B-1-F	0.3	5	2	100.41	17178+num of training samples
DENFIS	B-1-F	0.29	2	2	*	211043

*: the result was obtained under different computer environment, S: Sequential, B: Batch, 1: type-1, 2: Type 2, R: Recurrent, F: Feed-forward

B. Example 2: Tool Wear Prediction of Ball Nose End-Milling Process

This study case presents a tool wear prediction of a ball nose end milling process (Courtesy of Dr. Li Xiang, Singapore). It is a challenging problem, because the use of multi-point cutting tools at high speed, varying machining parameters, and inconsistency and variability of cutter geometry/dimensions makes accurate prediction extremely difficult to be achieved. A CNC milling process (Rőders Tech RFM760) with a spindle rate up to 42000 RPM was used during the experiment. Raw data were captured using seven channels DAQ, where the first three channels measured the force signal in three dimensional cutting axes, the next three channels collected the vibration signal in the three dimensional cutting axes, and the last channel recorded the acoustic emission (AE) signal. The prediction of the tool wear was carried out using the force signal from two different cutter profiles, where twelve input features were extracted. Force signal was selected, because it provides the most informative features of the tool wear prediction, while the tool wear itself was determined from visual inspection of flank wear utilising Olympus SZX16 microscope. A total of 630 data points were generated and the experiment was executed using 10-fold cross validation (CV) technique. The eT2RFNN was benchmarked against state-of-the art EFSs: PANFIS [3] , GENEFIS [37], simp_eTS [2], eTS [1], BARTFIS [40], eT2Class [17]. In addition, the eT2RFNN was run in several learning configurations as done in example 1 but without the

eT2RFNN(E), because the input-output (I/O) relationship is defined with the previous step observation only without lagged input attributes. Numerical results are tabulated in Table 3.

Table 3. Tool condition monitoring problem of a complex manufacturing process

Model	Type	RMSE	Rule	Input	Runtime	Parameters
eT2RFNN	S-2-R	**0.04±0.01**	**3.8±0.6**	**6.5±0.1**	**0.55±0.43**	**242.1**
eT2RFNN (A)	S-2-R	0.07±0.01	5.4±0.1	6.9±0.2	0.76±0.35	424.8
eT2RFNN(B)	S-2-R	0.06±0.03	**3.8±0.7**	12	1.1±0.8	820.8
eT2RFNN (C)	S-2-R	0.05±0.02	4.3±0.2	6.8±0.05	0.6±0.4	320.9
eT2RFNN(C1)	S-2-R	0.06±0.02	**3.8±0.8**	6.7±0.1	0.58±0.2	263.1
eT2RFNN(D)	S-1-R	0.11±0.01	5	7.5±0.3	0.9±0.6	485.75
eT2Class	S-2-R	0.05±0.01	5.9±01	12	1.1±0.7	2065
PANFIS	S-1-F	0.045±0.01	2	12	0.98±0.11	636.7
GENEFIS	S-1-F	0.0418±0.008	4.1±0.94	11.1±0.31	1.12±0.11	636.7
Simp_eTS	S-1-F	0.26±0.08	5	12	1.4±0.3	137
eTS	S-1-F	0.046±0.01	5.1±0.32	12	1.3±0.02	139.5
BARTFIS	S-1-F	0.0632±0.01	20.6±4.1	12	1.43±0.06	762.2
FAOSPFNN	B-1-F	0.27±0.003	12.7±0.7	12	1.91±0.1	330.2+num of training samples

*: the result was obtained under different computer environment, S: Sequential, B: Batch, 1: type-1, 2: Type 2, R: Recurrent, F: Feed-forward

The eT2RFNN (A)-(E) outperformed its counterparts in attaining tradeoff between complexity and accuracy. Although the eT2RFNN is built upon a generalized interval type-2 structure, the eT2RFNN possessed comparable number of parameters with other consolidated algorithms. This aspect is evident that the eT2RFNN network architecture is capable of suppressing fuzzy rule demand to low level. Furthermore, the feature selection mechanism of the eT2RFNN is capable of reducing the input dimension, thereby lowering network parameters to be saved in the memory. The efficacy of the double self-feedback loops is confirmed, where the learning performance dropped, when the second recurrent link was removed as depicted in the eT2RFNN(C),(C1).

C. Example 3: S&P 500 Index Time Series

This section focuses on evaluating the learning performance of eT2RFNN using the real-world financial time-series data. This study case is worth attempting, because the S&P 500 index time series data characterise dynamic and volatile behaviours, making them hard to deal with, especially for a non-evolving algorithm. 60 years of daily index values were collected from the Yahoo finance website and refer to the period from January 3, 1950 to March 12, 2009. In total, 14893 data streams were generated for the experiment, where all of which were fed in the training process, whereas reverse data were exploited to test the model's generalisation. As the dynamic of the real-world financial data stream, this problem is highly volatile and the most complex part lies after 1980. It is non-uniformly distributed in the interval of [16.66,1565.15]. eT2RFNN is benchmarked with 9 state-of-the art algorithms: GENEFIS [37], PANFIS [3], BARTFIS [40], ANFIS [42], eTS

[1], simp_eTS [2], DFNN [43], GDFNN [28], FAOSPFNN [43], eT2Class [17].We also tested some configurations of eT2RFNN to delineate the merits of each learning constituents as done in the example 1. Fig. 2(a) displays the predictive trend of eT2RFNN and the numerical results are encapsulated in Table 4. The I/O relationship of the model is governed as $y(n+1) = f(y(n-4), y(n-3), y(n-2), y(n-1), y(n))$. We just make use of $y(n+1) = f(y(n))$ for eT2RFNN(E) to exhibit the sensitivity to the delayed input feature, but we also provide the result exploiting all input attributes to disclose the efficacy of the online feature selection scenario. Fig. 2(b) visualises the trace of input attributes. The numerical results are reported in Table 4.

Table 4. S&P 500 Index Time Series

Model	Type	NDEI	Rule	Input	Runtime	Parameters
eT2RFNN	S-2-R	0.03	**2**	2	6.22	32
eT2RFNN (A)	S-2-R	0.04	5	2	8.33	80
eT2RFNN(B)	S-2-R	0.04	**2**	5	7.26	110
eT2RFNN(C)	S-2-R	0.04	4	2	6.6	52
eT2RFNN (C1)	S-2-R	0.05	**2**	2	6.2	26
eT2RFNN(D)	S-1-R	0.5	13	3	8.77	312
eT2RFNN(E)	S-2-R	**0.01**	**2**	**1**	**4.82**	**14**
eT2Class	S-2-F	0.05	5	5	21.6	385
PANFIS	S-1-F	0.09	4	5	55.3	144
GENEFIS	S-1-F	0.07	2	5	48.3	72
Simp_eTS	S-1-F	0.04	7	5	158.00	39
eTS	S-1-F	0.04	14	5	89.9	75
BARTFIS	S-1-F	0.02	8	5	12.3	128
DFNN	B-1-F	0.06	5	5	548.5	60+ num of training samples
GDFNN	B-1-F	0.07	4	5	951.4	64+ num of training samples
FAOSPFNN	B-1-F	0.07	13	5	159.8	91+num of training samples
ANFIS	B-1-F	0.02	32	5	384.9	222+num of training samples

S: Sequential, B: Batch, 1: type-1, 2: Type 2, R: Recurrent, F: Feed-forward

From Table 4, it can be observed that the eT2RFNN(E) demonstrated the most encouraging performance in all criteria. More interestingly, eT2RFNN(E) also incurred the least network parameters as a result of its generalised fuzzy rule, which can diminish the fuzzy rule demand, and its double recurrent links, which can overcome the absence of delayed input features. The network simplicity was achieved without any loss of accuracy and expedites the training process. The difference between the batched and sequential algorithms is also noticeable, where the batched algorithm imposes intractable computational load. In Fig.2(a), the efficacy of the online feature selection is confirmed, where it automatically reduces the dimensionality of input space without undermining the model's generalisation.

D. Example 4: SISO Dynamic System Identification

This numerical example aims to compare the effectiveness of eT2RFNN's learning performance with state-of-the art RFNNs in the popular benchmark problem, the SISO dynamic

system identification. This problem is commonly used to delve the performance of RFNNs, because it characterises the temporal system dynamic as a result of the temporal control input. In short, the mathematical model of the nonlinear system is described as follows:

$$y(n+1) = 0.72y_p(n) + 0.025y(n-1)u(n-1) + 0.01u^2(n-2) + 0.2u(n-3) \quad (58)$$

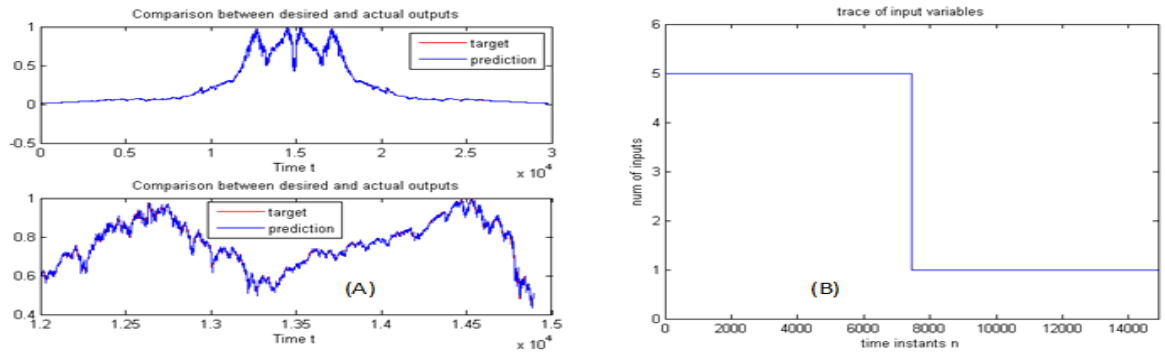

Fig.2: (a) eT2RFNN's prediction in S&P 500 time series, (b) the trace of eT2RFNN's input attribute in S&P 500 time series

Table 4. SISO dynamic system identification

Model	Type	RMSE	Rule	Input	Runtime	Parameters
eT2RFNN	S-2-R	**0.005**	**2**	**1**	0.34	14
eT2RFNN (A)	S-2-R	0.05	4	**1**	0.5	28
eT2RFNN(B)	S-2-R	0.07	**2**	2	0.47	32
eT2RFNN(C)	S-2-R	0.05	4	**1**	0.5	22
eT2RFNN (C1)	S-2-R	0.18	**2**	**1**	**0.3**	**11**
eT2RFNN(D)	S-1-R	0.04	5	2	1.1	70
eT2Class	S-2-F	0.01	4	2	1.1	80
RSEIT2FNN	S-2-R	0.006	2	2	*	18
RSEFNN-LF	S-1-R	0.03	4	2	*	30
MRIT2NFS	S-2-R	0.007	5	2	*	40
SLFRWNN	S-1-R	0.008	7	2	*	33+num of training samples

*: the result was obtained under different computer environment, S: Sequential, B: Batch, 1: type-1, 2: Type 2, R: Recurrent, F: Feed-forward

where $y(n), u(n)$ stand for the system output and the control input respectively. The input and output relationship of the regression model was $y(n+1) = f(y(n), u(n))$, which excludes any lagged input features. 900 data points were generated to train the network, where the control input were sampled from the uniformly distributed random sequence in the range of [-2,2] for the first half of 900 samples, whereas the control input for the remaining time instants was generated from the sinusoidal function $1.05\sin(\pi/45)$. The identification problem was done using two input attributes *y(n),u(n)* and the target variable was *y(n+1).* The validation phase was carried out using 1000 samples, in which $u(n)$ is set as a temporal input variable as follows:

$$u(n) = \begin{cases} \sin(\pi/25), n < 250 \\ 1, 250 \le n < 500 \\ -1, 500 \le n < 750 \\ 0.3\sin(\pi/25) + 0.1\sin(\pi/32) + 0.6\sin(\pi/10), 750 \le n < 1000 \end{cases} \quad (59)$$

eT2RFNN was compared with 4 state-of-the art RFNNs recently published in the literature: RSEIT2FNN [16], RSEFNN-LF [12], SLFRWNN [18], MRIT2NFS [36], eT2Class [17]. SLFRWNN constitutes a recurrent type-1 fuzzy wavelet neural network, which incorporates the local recurrent link in the rule consequent, while the RSEFNN-LF presents the type-1 fuzzy neural network with a local feedback loop in the rule layer. The RSEIT2FNN forms an extended version of the RSEFNN-LF in the interval type-2 fuzzy environment, while the MRIT2NFS actualizes a recurrent interval type-2 fuzzy network with the interactive recurrent network architecture. By extension, eT2RFNN was run in different settings as detailed in the example 1 but the eT2RFNN(E) was ruled out, because the standard setting of this problem solely relies on the most recent observation *n* to perform one-step ahead prediction. The results are served in Table 5.

Table 5. Learning Characteristic of Consolidated Algorithms

Algorithm	Recurrent types	Premise Part	Consequent Part	Learning Type	Learning Modules
eT2RFNN	Double Local	Interval type-2 multivariate Gaussian	Nonlinear Wavelet	Online	RG+RA+RP+RM-RC+FS
RSEIT2FNN	Single Local	Interval type-2 univariable Gaussian	First order TSK	Online	RG+RA
MRIT2NFS	Single Interactive	Interval type-2 univariable Gaussian	First order TSK	Online	RG+RA
TRFNS	Single Local	Type-1 univariable Gaussian	First order TSK	Offline	RG+RA
RSEFNN-LF	Single Local	Type-1 univariable Gaussian	First order TSK	Online	RG+RA
SLFRWNN	Single Local	Type-1 univariable Gaussian	Nonlinear Wavelet	Offline	RG+RA

RG: Rule Growing, RA: Rule Adaptation, RP: Rule Pruning, RM: Rule Merging, RC: Rule Recall, FS: Feature Selection

From Table 5, eT2RFNN produced the most reliable accuracy, while sustaining the most frugal computational and structural complexities. Furthermore, it is evident that the online dimensionality reduction method not only functions to ease the network burden, but also is capable of boosting the model's generalisation. The absence of the online feature selection method as shown in eT2RFNN(b) degenerated the predictive accuracy. The learning performance of eT2RNN substantially deteriorated when realised in the conventional recurrent type-2 FNN as used in RSEIT2FNN. This situation justifies the new eT2RFNN network topology plays critical role for the training process. Although the SLFRWNN was inferior to the eT2RFNN, its efficacy is noticeable in Table 5, where it delivered competitive numerical results against interval type-2 FNNs: MRIT2NFS, RSEIT2FNN.

VI. Conceptual Comparison

The efficacy of eT2RFNN has been experimentally validated using five non-stationary data streams and comparisons with prominent FNNs in Section 4. This section specifically analyses the key facets of eT2RFNN, which distinguish itself with its counterpart. To this end, eT2RFNN is conceptually contrasted with six popular RFNNs in the literature: SLFRWNN [18], TRFNS [13], RSEFNN-LF [12], RSEIT2FNN [16], MRIT2NFS [36]. The characteristics of the consolidated algorithms are tabulated in Table 6.

Referring to Table 6, eT2RFNN goes one step ahead of existing RFNNs by putting forward double local recurrent connections. This architecture retains the local learning trait, offering greater flexibility and robustness for the evolving system. eT2RFNN employs a more advanced type of fuzzy rule, which synergises the interval type-2 multivariate Gaussian function in the rule premise and the Wavelet function. Most other algorithms in Table 7 except RSEFNN-LF either utilise the KM iterative method or the GA method. The GA method is renowned for its offline trait, because it relies on the multi-pass optimization process. Although the KM method is applicable for an online learning process, it is deemed computationally more expensive than the *q*-design factor of the eT2RFNN, because it requires the weight vector to be reordered in the ascending manner. The crossover points are then found using the iterative process. Other RFNNs in Table 7 are presumed to impose more prohibitive structural complexity, because they are not fitted out by a rule base simplification procedure.

VII. Conclusion and Further Study

A novel recurrent interval type-2 FNN, namely Evolving Type-2 Fuzzy Neural Network (eT2RFNN), is proposed with research question of handling concept drifts and uncertainties in data stream mining. The eT2RFNN presents a fully flexible and computationally efficient working principle, which can adapt, grow, prune, and merge its fuzzy rules and even reduce the dimensionality of learning problems on the fly. The salient contributions of eT2RFNN are summed up as follows: 1) It is implemented in a novel recurrent network architecture, 2) It puts forward a generalised interval type-2 fuzzy rule, 3) the universal approximation capability of the eT2RFNN's network architecture is mathematically proven, 4) A type-2 version of the RMI method is developed, and 5) the online feature selection technique, namely the SMBC method, is proposed.

Our numerical study in various synthetic and real-world data problems demonstrated that eT2RFNN outperformed state-of-the art algorithms in both complexity and accuracy. Our future work will be devoted to designing a metacognitive scaffolding learning algorithm for eT2RFNN.

APPENDIX

A. *Universal Approximation Property of the eT2RFNN*

The universal approximation property of the eT2RFNN can be achieved by extending the work of [59] for the single-recurrent-link network architecture to the double-recurrent-link case. An open dynamical system in discrete time can defined as a state transition and an output equation:

$$s_{n+1} = g_1(s_n, u_n) \text{ (state transition)}, \; y_n = h(s_1) \text{ (output equation)} \tag{A.1}$$

where *s* stands for state transition, a mapping from the present internal hidden state of the system and the external input u_n to a new system state. Note that if the state transition is discounted, we revover the expression of the feedforward network architecture, whereas without the external input u_n we arrive at the definition of the autonomous system. The predictive task however will be harder with the hidden state, because it takes into account inter temporal dependencies. In [59], a recurrent neural network in the state space form is developed to map an open dynamical system (A.1). The expression is amended here to suit the eT2RFNN structure as follows:

$$S_1^{n+1} = \tilde{f}(A_1 S_1^n + B_1 U_n), S_2^{n+1} = \tilde{f}(A_2 S_2^n + B_2 U_n) \text{ (State Transitition)} \tag{A.2}$$

$$Y_n = C_1 C_2 S_1^{n+1} S_2^{n+1} \text{ (output equation)} \tag{A.3}$$

where $A_{1,2}, B_{1,2}, C_{1,2}$ are weight metrices of appropriate dimension and $\tilde{f}()$ is an interval type-2 fuzzy set. The state space form eases the analysis because of explicit correspondence between equations and architecture, from which it can be directly transferred into a spatial network structure via the so-called unfolding in time and shared weight matrices $A_{1,2}, B_{1,2}, C_{1,2}$.

Theorem 2 [59]: let $g : \Re^P \times \Re^u \to \Re^m$ be measurable and $h : \Re^P \to \Re^m$ be continuous, the external input $u_n \in \Re^u$, the inner state $s_n \in \Re^P$, and the output $y_n \in \Re^m$. Then, an open dynamical system, defined in (A.1)can be predicted by an element of the RNN (A.2)-(A.3) with an arbitrary accuracy, where $\tilde{f}()$ is an interval type-2 fuzzy set.

Proof : the mathematical proof can be obtained by tracing back the equations of the open dynamical system with the representation of the feedforward network structure as a result of

transforming the RNN definition (A.2)-(A.3) into its feedforward network structure equivalent, using the unfolding in time and shared weigth matrices. This can be achieved in two steps: 1) a conclusion of the state space equation of the open dynamical system can be approximated by a neural network in the form of (A.2); 2) the output equation of the open dynamical system can be approximated by a neural network of the form of (A.3). Detailed mathematical derivations are left to reader and it can be achieved following the same procedure of [59].

ACKOWLEDGEMENT

The work presented in this paper is supported by the Australian Research Council (ARC) under Discovery Project DP150101645 and the Latrobe university start-up grant. The third author acknowledges the support of the Austrian COMET-K2 program of the Linz Center of Mechatronics (LCM), funded by the Austrian federal government and the federal state of Upper Austria.

REFERENCES

[1] P.Angelov and D. Filev, "An approach to online identification of Takagi-Sugeno fuzzy models," IEEE Transactions on Systems, Man, and Cybernetics, Part B.vol. 34, pp. 484-498.(2004)
[2] P.Angelov and D. Filev, "Simpl_eTS: A simplified method for learning evolving Takagi-Sugeno fuzzy models," in IEEE International Conference on Fuzzy Systems (FUZZ), pp. 1068-1073.(2005)
[3] M. Pratama, S. Anavatti, P. Angelov, E. Lughofer, PANFIS: A Novel Incremental Learning, IEEE Transactions on Neural Networks and Learning Systems, Vol.25, no.1, pp.55-68,(2014)
[4] N. N. Karnik, J. M. Mendel, and Q. Liang, "Type-2 fuzzy logic systems, IEEE Transactions on Fuzzy Systems, vol. 7, no. 6, pp. 643–658, (1999)
[5] L. A. Zadeh, "The concept of a linguistic variable and its application to approximate reasoning," Information Sciences., vol. 8, no. 3, pp. 199–249, (1975)
[6] Q. Liang and J. M. Mendel, "Interval type-2 fuzzy logic systems: Theory and design," IEEE Transactions on Fuzzy Systems, vol. 8, no. 5, pp. 535–550, (2000)
[7] C. F. Juang and Y. W. Tsao, "A self-evolving interval type-2 fuzzy neural network with online structure and parameter learning," IEEE Transactions on Fuzzy Systems, vol. 16, no. 6, pp. 1411–1424, (2008)
[8] J. M. Mendel, "Computing derivatives in interval type-2 fuzzy logic system," IEEE Transactions on Fuzzy Systems, vol. 12, no. 1, pp. 84–98, (2004)
[9] R. H. Abiyev and O. Kaynak, "Type-2 fuzzy neural structure for identification and control of time-varying plants," IEEE Transactions on Industrial Electronics , vol. 57, no. 12, pp. 4147–4159,(2010)
[10] Y. Y. Lin, J. Y. Chang, and C. T. Lin, "A TSK-type based self-evolving compensatory interval type-2 fuzzy neural network (TSCIT2FNN) and its applications," IEEE Transactions on Industrial Electronics., vol. 61, no. 1, pp. 447–459, (2014)
[11] C.F. Juang and C.T. Lin, "A recurrent self-organizing neural fuzzy inference network," IEEE Transactions on Neural Networks, vol.10, pp.828-845, (1999)
[12] C. F. Juang, Y. Y. Lin, and C. C. Tu, "A recurrent self-evolving fuzzy neural network with local feedbacks and its application to dynamic system processing," Fuzzy Sets and Systems, vol.161,no.19, (2010)
[13] C.F. Juang, "A TSK-type recurrent fuzzy network for dynamic systems processing by neural network and genetic algorithms," IEEE Transactions on Fuzzy Systems, vol.10,no.2, pp.155-170, (2002)
[14] Yang-Yin Lin, Jyh-Yeong Chang, Chin-Teng Lin," Identification and Prediction of Dynamic Systems Using an Interactively Recurrent Self-Evolving Fuzzy Neural Network", IEEE Transactions on Neural Networks and Learning Systems, Vol.24,no.2,pp.310-321, (2013)
[15] M.Pratama, J.Lu, S.Anavatti," Recurrent Classifier based on An Incremental Meta-Cognitive-based Scaffolding Algorithm", online and in press, IEEE Transactions on Fuzzy Systems, (2015)
[16] C. F. Juang, R. B. Huang and Y. Y. Lin, "A recurrent self-evolving interval type-2 fuzzy neural network for dynamic system processing," IEEE Transactions on Fuzzy Systems, vol.17, no.5, pp.1092-1105, (2009)
[17] M. Pratama, J. Lu, G.Zhang, " Evolving Type-2 Fuzzy Classifier", online and in press, IEEE Transactions on Fuzzy Systems, on line and in press, (2015)
[18] S. Ganjefar, M. Tofighi, "Single-hidden-layer fuzzy recurrent wavelet neural network: Applications to function approximation and system identification", Information Sciences, online and in press, (2014)
[19] H. Gan, et al, "Nonlinear Systems Modeling Based on Self-Organizing Fuzzy-Neural-Network with Adaptive Computation Algorithm ", IEEE Transactions on Cybernetics, Vol. 44(4), pp.554-564, (2014)
[20] P. Angelov, Evolving Takagi-Sugeno Fuzzy Systems from Data Streams (eTS+), In Evolving Intelligent Systems: Methodology and Applications (Angelov P., D. Filev, N. Kasabov Eds.), John Willey and Sons, IEEE Press Series on Computational Intelligence, pp. 21-50, ISBN: 978-0-470-28719-4, April 2010
[21] E.Lughofer,"On-line incremental feature weighting in evolving fuzzy classifiers," Fuzzy Sets and Systems, vol. 163(1), pp. 1–23, (2011)
[22] L. Yu, H. Liu, "Efficient Feature Selection via Analysis of Relevance and Redundancy", Journal of Machine Learning Research, Vol.5, pp.1205-1224, (2004)
[23] R.H, Abiyev, O. Kaynak, " Fuzzy Wavelet Neural Networks for Identification and Control of Dynamic Plants-A Novel Structure and a Comparative Study", IEEE Transactions on Industrial Electronics,vol.55(8), pp.3133-3140, (2008)
[24] J.C.Patra, A.C.Kot, " Nonlinear dynamic system identification using Chebyshev functional link artificial neural networks", IEEE Transactions on Systems, Man and Cybernetics-Part B: Cybernetics, Vol.32,no.4, pp.505-511, (2002)
[25]R.E. Moore, R.B. Kearfott, M.J.Cloud. Introduction to Interval Analysis. SIAM – Philadelphia, (2009).

[26] M.Pratama, S.Anavatti, E.Lughofer, " pClass:An Effective Classifier to Streaming Examples", IEEE Transactions on Fuzzy Systems, Vol.23(2), pp.369-386, (2014)
[27] L.Wang, H-B.Ji, Y.Jin, " Fuzzy Passive-Aggressive Classification: A Robust and Efficient Algorithm for Online Classification Problems", Information Sciences, Vol.220,pp.46-63, (2013)
[28] S.-Q. Wu, M-J. Er, and Y. Gao, A fast approach for automatic generation of fuzzy rules by generalized dynamic fuzzy neural networks, IEEE Transaction on Fuzzy System,Vol.9(4),pp.578–594,(2003).
[29] E. Lughofer, Evolving Fuzzy Systems --- Methodologies, Advanced Concepts and Applications, Springer, Heidelberg, 2011
[30] B. Vigdor and B. Lerner, "The Bayesian ARTMAP," IEEE Transactions on Neural Networks, vol. 18, no. 6, pp. 1628–1644, (2007)
[31] P. Mitra, C.A. Murthy, S.K. Pal," Unsupervised feature selection using feature similarity", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.24(3), pp.301-312, (2002)
[32] R.J. Oentaryo, M. Pasquier, C. Quek, "RFCMAC: A novel reduced localized neuro-fuzzy system approach to knowledge extraction", Expert Systems with Applications, Vol.38, pp.12066-12084, (2011)
[33] A. Lazo, et al," On the entropy of continuous probability distributions", IEEE Transactions on Information Theory, 24(1), 120–122, (1978)
[34] C-F.Juang, C-Y.Chen, "Data-Driven Interval Type-2 Neural Fuzzy System With High Learning Accuracy and Improved Model Interpretability", IEEE Transactions on Cybernetics, Vol.43, no.6, pp.1781-1795, (2013)
[35] S.W.Tung, C.Quek, C.Guan, "eT2FIS: An Evolving Type-2 Neural Fuzzy Inference System", Information Sciences, vol.220, pp.124-148, (2013)
[36] Y-Y. Lin, J-Y. Chang, N. R. Pal, C-T. Lin, " A Mutually Recurrent Interval Type-2 Neural Fuzzy System (MRIT2NFS) With Self-Evolving Structure and Parameters", *IEEE Transactions on Fuzzy Systems*, Vol.21(3), pp.492-509, (2013)
[37] M.Pratama, S.Anavatti, E.Lughofer, " GENEFIS:Towards An Effective Localist Network", IEEE Transactions on Fuzzy Systems, Vol.25, no.3, pp.547-562, (2014)
[38] P. Angelov, Ronald R. Yager: A new type of simplified fuzzy rule-based system. International Journal of General Systems vol.41(2),pp. 163-185 (2012)
[39] N. Kasabov, and Q. Song, DENFIS: dynamic evolving neural-fuzzy inference system and its application for time series prediction, IEEE Transactions on Fuzzy Systems .vol10 (2).pp. 144–154. (2002)
[40] R. J. Oentaryo, M. J. Er, S. Linn, and X. Li, "Online probabilistic learning for fuzzy inference systems," Expert Systems with Applications, vol. 41, no. 11, pp. 5082-5096, (2014)
[41] J.-S. R. Jang, "ANFIS: Adaptive-network-based fuzzy inference system", IEEE Transaction on System. Man. Cybernetic, part b: cybernetics, vol. 23, pp. 665–684,(1993)
[42] N.Wang, M.J.Er, and M.X, Fast and Accurate Self Organizing Scheme for Parsimonious Fuzzy Neural Network, Neurocomputing, Vol 72,(2009)
[43] S. Wu, and M-J. Er, Dynamic fuzzy neural networks—a novel approach to function approximation, IEEE Transaction on Systems Man Cybernetics, part b: Cybernetics,vol.30,pp. 358–364,(2000)
[44]O. Castillo, J.R. Castro, P. Melin, A. Rodriguez-Diaz, " Universal Approximation of a Class of Interval Type-2 Fuzzy Neural Networks in Nonlinear Identification", Advances in Fuzzy Systems, Vol.2013(7), (2013)
[44] J. Gama, Knowledge Discovery from Data Streams, Chapman & Hall/CRC, Boca Raton, Florida, 2010
[45] M. Sayed-Mouchaweh and E. Lughofer, Learning in Non-Stationary Environments: Methods and Applications, Springer, New York, 2012
[46] A. Zdsar, D. Dovzan and I. Skrjanc, Self-tuning of 2 DOF control based on evolving fuzzy model, *Applied Soft Computing*, vol. 19, pp. 403--418, 2014
[47] E. Lughofer, On-line Assurance of Interpretability Criteria in Evolving Fuzzy Systems --- Achievements, New Concepts and Open Issues, *Information Sciences*, vol. 251, pp. 22-46, 2013
[48] E. Lughofer, C. Cernuda, S. Kindermann and M. Pratama, Generalized Smart Evolving Fuzzy Systems, *Evolving Systems*, on-line and in press, doi: 10.1007/s12530-015-9132-6, 2015
[49] E. Lughofer and M. Sayed-Mouchaweh, Autonomous Data Stream Clustering implementing Incremental Split-and-Merge Techniques --- Towards a Plug-and-Play Approach, Information Sciences, vol. 204, pp. 54-79, 2015
[50] Juan R. Castro, Oscar Castillo, Patricia Melin, Antonio Rodríguez Díaz: A hybrid learning algorithm for a class of interval type-2 fuzzy neural networks. Information Sciences, Vol.179(13), pp.2175-2193, 2009
[51] Patricia Melin, Claudia I. González, Juan R. Castro, Olivia Mendoza, Oscar Castillo: Edge-Detection Method for Image Processing Based on Generalized Type-2 Fuzzy Logic. IEEE Transactions on Fuzzy Systems, Vol.22(6), pp. 1515-1525, 2014
[52] Oscar Castillo, Juan R. Castro, Patricia Melin, Antonio Rodríguez Díaz: Application of interval type-2 fuzzy neural networks in non-linear identification and time series prediction. Soft Computing, Vol.18(6), pp.1213-1224, 2014
[53] Hagras,H., Doctor,F.,Callaghan, V., Lopez, A. An Incremental Adaptive Life Long Learning Approach for Type-2 Fuzzy Embedded Agents in Ambient Intelligent Environments. IEEE Transactions on Fuzzy Systems, Vol.15(1), pp.41-55, 2007
[54] Jammeh, E.A., Fleury, M., Wagner, C., Hagras, H., Ghanbari, M. Interval Type-2 Fuzzy Logic Congestion Control for Video Streaming Across IP Networks. IEEE Transactions on Fuzzy Systems, Vol.17(5), 1123-1142, (2009)
[55]P. P. Angelov, Evolving Rule-based Models: A Tool for Design of Flexible Adaptive Systems, Springer-Verlag, Heidelberg, New York, 2002, ISBN 3-7908-1457-1
[56] S, Chiu, "Fuzzy Model Identification Based on Cluster Estimation," Journal of Intelligent & Fuzzy Systems, Vol. 2(3), 1994
[57] R.R, Yager, D.P, Filev, " Approximate clustering via the mountain method", IEEE Transactions Systems, Man and Cybernetics, Vol.24(8), pp. 1279-1284, 1994
[58] Witten, I.H., Frank, E. and Hall, M.A. (2011) Data Mining: Practical machine learning tools and techniques. (Third Edition). Morgan Kaufmann, USA
[59] A. M, Schafer, H-G, Zimmermann, " Recurrent Neural Networks are Universal Approximators", International Journal of Neural Systems, Vol.7(4), (2007)
[60] E. Lughofer, V. Macian, C. Guardiola and E.P Klement, " Identifying Static and Dynamic Prediction Models for NOx Emissions with Evolving Fuzzy Systems", Applied Soft Computing, vol. 11(2), pp. 2487-2500, 2011
[61] M. Pratama, M. J. Er, X. Li, R. J. Oentaryo, E. Lughofer, and I. Arifin, "Data driven modeling based on dynamic parsimonious fuzzy neural network," Neurocomputing, vol. 110, pp. 18-28, 2013
[62] P. Angelov, "Fuzzily connected multimodel systems evolving autonomously from data streams," IEEE Transactions on Systems, Man, and Cybernernetics, Part B: Cybernetics, vol. 40(4), pp. 898–910, 2010.