On-line Redundancy Elimination in Evolving Fuzzy Regression Models using a Fuzzy Inclusion Measure *

Edwin Lughofer[1] **Eyke Hüllermeier**[2]

[1]Department of Knowledge-Based Mathematical Systems, Johannes Kepler University, A-4040 Linz, Austria
[2]Department of Mathematics and Computer Science, Philipps-Universität Marburg, Germany

Abstract

This paper tackles the problem of complexity reduction in evolving fuzzy regression models of the Takagi-Sugeno type. The incremental model adaptation process used to evolve such models over time, often produces redundancies such as overlapping rule antecedents. We propose the use of a fuzzy inclusion measure in order to detect such redundancies as well as a procedure for merging rules that are sufficiently similar. Experimental studies with two high-dimensional real-world data sets provide evidence for the effectiveness of our approach; it turns out that a reduction in complexity is even accompanied by an increase in predictive accuracy.

Keywords: evolving fuzzy models, incremental learning, regression, fuzzy inclusion, rule merging, fuzzy set merging, complexity reduction

1. Introduction

In nowadays industrial systems, there is an increasing demand of automatic model updates as new upcoming operating conditions, system behaviors [1], new types of classes [2] or even drift occurrences [3] may arise during on-line processes. These situations should be included into the models in order to guarantee robust and process-save operations (predictions, control behaviors etc.) [4] [5]. Therefore, Evolving fuzzy systems (EFS) have received increasing attention in the recent years [6]. Such systems allow for producing and maintaining fuzzy (rule-based) models in a data-driven way, which is accomplished by learning and adapting these models in an on-line, incremental manner on a continuous stream of data [7]. Examples of EFS algorithms are the *DENFIS* approach (short for *Dynamic Evolving Neural Fuzzy-Inference System*) [8], *eTS* (short for *evolving Takagi-Sugeno fuzzy systems*) and its extended version *eTS+* [5], *ePL* (short for *evolving Participatory Learning*) [9] or *SAFIS* (short for *Sequential Adaptive Fuzzy Inference Systems*) [10].

Due to the incremental, local learning process, different types of redundancies might be produced for EFS in the course of time. In particular, since existing rules can be moved in the input space and new rules can be created, there is a danger of producing highly overlapping and hence partly redundant rules. As a consequence, one may obtain unnecessarily complex systems with an ever increasing number of rules and fuzzy sets.

Research on evolving fuzzy systems so far has mainly focused on the predictive accuracy of the models produced, while less attention has been paid to the issue of model complexity. There are a few exception, though. In [11], the authors use a geometric similarity measure for detecting redundant fuzzy sets and a weighted average for merging the parameters of redundant (Gaussian) fuzzy sets and rule consequents. The idea of removing obsolete rules with low support by past samples was presented in [12] and, based on the concepts of *rule age* and *rule utility*, further extended in [5]. In [9], rules correspond to clusters in the input space, and redundant rules are detected by calculating the sum of absolute deviations between the (normalized) coordinates of two cluster centers, whereas the original cluster center is maintained. Another approach for removing redundant fuzzy sets is proposed in [13], where specific properties of Gaussian fuzzy sets are exploited.

Building on the latter approach and adopting ideas from [11], this paper improves upon existing approaches in several respects. More specifically, we make the following contributions to evolving fuzzy regression models of the Takagi-Sugeno (TS) type:

1. Redundancy of rules is detected in the high-dimensional feature space according to a fuzzy inclusion measure determining the degree of inclusion of a rule A in a rule B. The same measure is used to reduce redundancies on the level of one-dimensional fuzzy partitions (produced by projecting rules to the individual axes).
2. The merging of two redundant multi-dimensional rules, each one represented by a cluster in the input space, is performed by a novel cluster merging procedure taking the significance of clusters into account and using a variance update formula for properly modulating the range of influence of the newly created rule (cluster). A similar procedure is proposed for merging fuzzy sets.
3. A novel concept for a consistent treatment of contradictory rules is proposed.

*This work was funded by the Austrian fund for promoting scientific research (FWF, contract number I328-N23, acronym IREFS). This publication reflects only the authors' views.

In the next section, we recall some basic concepts of evolving fuzzy systems. In Section 3, we introduce methods for detecting and eliminating significant overlap (redundancy) of rules in the high-dimensional feature space and of fuzzy sets on the level of one-dimensional fuzzy partitions. Experimental results are presented in Section 4, where *FLEXFIS* will be used as EFS learning engine and serving a concrete implementation of our ideas (though other methods built on the same fuzzy systems architecture would serve the same purpose).

2. Evolving Fuzzy Systems for Regression

The most commonly used model architecture in modern EFS for regression, is the Takagi-Sugeno (TS) fuzzy systems architecture:

$$\hat{f}(\vec{x}) = \hat{y} = \sum_{i=1}^{C} l_i \Psi_i(\vec{x}) \qquad (1)$$

with the normalized membership functions

$$\Psi_i(\vec{x}) = \frac{\mu_i(\vec{x})}{\sum_{j=1}^{C} \mu_j(\vec{x})}, \quad \mu_i(\vec{x}) = \prod_{j=1}^{p} \mu_{ij}(x_j) \qquad (2)$$

and consequent functions

$$l_i = w_{i0} + w_{i1}x_1 + w_{i2}x_2 + ... + w_{ip}x_p \ . \qquad (3)$$

Here, p is the dimensionality of the learning problem, and x_j the value of the j-th input variable. Moreover, μ_{ij} denotes the fuzzy set in the j-th antecedent of the i-th rule. These antecedents are combined by means of a t-norm [14].

Among various fuzzy systems architectures, the TS model is able to provide the most accurate estimates and is therefore often used for modeling tasks in which precision is of major importance. Besides, however, interpretability is an important criterion, too. Even though TS models are generally considered with reservation from this point of view, one can argue that an understanding of such models is in principle possible, especially when using local learning for the consequent parts [15]. In any case, it is hardly disputable that interpretability presumes a reasonable level of *complexity*. In this paper, we therefore propose different methods for reducing the *unnecessary* complexity of an EFS.

3. On-Line Complexity Reduction in EFS

This section deals with two approaches for reducing the complexity of evolving TS fuzzy systems based on local redundancy criteria. The first approach is directly applied in the cluster space (recall that each rule is associated with a cluster) and merges clusters which are strongly overlapping. The second approach acts on the fuzzy partition space and performs a merging of fuzzy sets for each dimension (variable) separately. Of course, since rules are defined in terms of these fuzzy sets, this can again have an influence on the redundancy of rules.

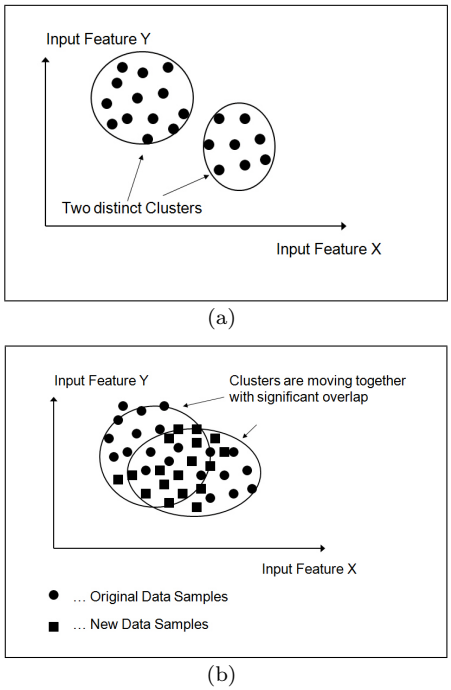


Figure 1: (a): Two distinct clusters (rules) in the two-dimensional feature space, indicated as ellipsoids; (b): significant overlap caused by an adaptation to new samples (rectangles) falling in-between these clusters.

We like to emphasize the following properties of both approaches: First, they both allow an incremental, *single-pass* processing of the data, which means that, in addition to the model itself, the update process only requires the new data sample, which is immediately discarded afterward. Second, both methods are completely general and can be used for any types of fuzzy sets.

3.1. Rule Merging in the Feature Space

Rules can become overlapping due to the fact that clusters are moving in the feature space. Imagine, for example, a two-dimensional feature space with two clusters that are initially disjoint (see Fig. 1). If new data points are emerging in-between these clusters, they may both grow in size, and their centers may move toward each other. Eventually, this may result in a significant overlap.

3.1.1. Measuring the Redundancy of Rules

In order to measure the redundancy of a rule A with respect to another rule B, we compute a fuzzy degree of inclusion of A in B. In fact, redundancy is obviously better reflected by inclusion than by similarity. Besides, whenever needed, similarity can be *derived* from the primitive notion of inclusion, namely as a kind of mutual inclusion: A and B

are similar if A is fuzzily included in B and, vice versa, B is fuzzily included in A. More specifically, we compute a standard measure of fuzzy inclusion subsequent to each incremental learning step:

$$INC(A,B) = \prod_{i=1}^{p} inc(A_i, B_i), \tag{4}$$

with A_i the fuzzy set in the i-th antecedent part and

$$inc(A_i, B_i) = \frac{\int min(A_i(x), B_i(x))dx}{\int B_i(x)dx} \tag{5}$$

the degree of inclusion of the fuzzy set A_i in B_i. Thus, A is included in B if a corresponding inclusion holds in all dimensions, i.e., for the projections of the two clusters to each of the axes. As a concrete t-norm, we shall use the minimum. Despite the fact that this measure needs some significant computation power for calculating the integral in (5), it is still faster than a direct calculation of the intersection degree between two ellipsoids in the high-dimensional space with complex mathematical formulas [16] ($O(pn)$ with n discretization steps of the integral versus $O(p^3)$).

For specific types of fuzzy sets, (5) can be derived analytically. Otherwise, it is always possible to make use of a suitable discretization, replacing the integral by a sum.

The overlap between two rules can then be defined as follows:

$$OL(A,B) = \perp(INC(A,B), INC(B,A)), \tag{6}$$

where $\perp$ is a t-conorm. Thus, there is an overlap between A and B if either A is included in B or B is included in A (note that a conjunction by using a t-norm at this place would model equality, which is a stricter condition and would not be able to resolve a full embedding of a smaller cluster in a larger one. As a concrete t-conorm, we shall use the maximum.

3.1.2. Merging Rule Antecedents

Two rules A and B are merged if their overlap degree (6) exceeds a threshold sim_{thr}. Assuming that a rule corresponds to a hyper-ellipsoid (e.g., when using Gaussians membership functions) or a hyper-box (e.g., when using trapezoidal or triangular functions) in the feature space, we denote by c_j^A the j-th center coordinate of rule A and by σ_j^A the range of influence in the j-th direction (main axis of hyper-ellipsoid or width of the hyper-box).

The merging itself is realized by computing a weighted average of the rules, with the weights being proportional to the number of number of samples k_A and k_B covered by the rules, respectively. Assuming $k_A \geq k_B$, we subsequently consider rule A as the more relevant and rule B the less relevant rule. A combination of the ranges of influence (widths) of the two rules is done by updating the range of influence (in each direction) of the rule A with the range of influence of rule B. To this end, we make use of the recursive variance formula [17], modified in an appropriate way (updating the range of influence of rule A using the center and range of influence of rule B). In order to guarantee a good coverage of the original data cloud by the new rule, a fraction of the variance of samples belonging to rule B is added, which is determined by the percentage of samples $k_B/(k_A + k_B)$. In summary, the two rules are merged as follows:

$$\begin{aligned} c_j^{new} &= \frac{c_j^A k_A + c_j^B k_B}{k_A + k_B} \\ \sigma_j^{new} &= \sqrt{\frac{k_A(\sigma_j^A)^2}{k_A + k_B} + (c_j^A - c_j^{new})^2 + \frac{(c_j^{new} - c_j^B)^2}{k_A + k_B}} \\ &\quad + \frac{k_B \sigma_j^B}{k_A + k_B} \\ k_{new} &= k_A + k_B \end{aligned} \tag{7}$$

Figure 2 demonstrates two examples where two rules (represented as clusters) are merged because of a high resp. medium overlap degree.

3.1.3. Merging Rule Consequents

Merging of two rule antecedents by (7) also necessitates a merging of the corresponding consequent parts. Again, this is accomplished by taking the weighted mean, with respective weights $k_A/(k_A + k_B)$ and $k_B/(k_A + k_B)$:

$$w_{new} = \frac{\vec{w}_A k_A + \vec{w}_B k_B}{k_A + k_B}, \tag{8}$$

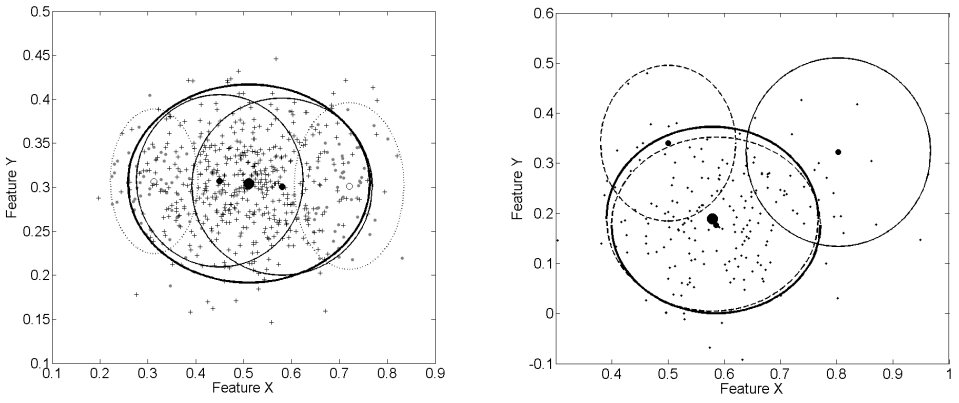
where $\vec{w}_A$ and $\vec{w}_B$ denote the vector of coefficients of the linear functions in the antecedent of rule A and B, respectively.

A high dissimilarity of the consequent functions in (8) may indicate an inconsistency in the rule base: Two rules with very similar condition part yield very different conclusions. In a case like this, a simple linear combination of the consequent parts may not appear appropriate.

In TS fuzzy systems, rule consequent functions are represented by hyper-planes in the output space. A reasonable measure for similarity, therefore, is the angle between these hyper-planes, as it measures the difference of the direction followed by the consequent functions in the output space. The angle between two hyper-planes (corresponding to the rules A and B) can be measured by the angle between the corresponding normal vectors a and b:

$$\phi = \arccos\left(\left|\frac{\vec{a}^T\vec{b}}{|\vec{a}||\vec{b}|}\right|\right) \in [0, \pi] \tag{9}$$

The maximal dissimilarity is obtained when the angle between the two normal vectors is $\frac{\pi}{2}$, as the orientation of the vectors do not play a role. The similarity of two hyper-planes y_A and y_B can then be



(a) (b)

Figure 2: (a): strongly overlapping clusters (overlap degree of 0.88) shown as thin solid lines are merged to one bigger cluster (thick solid line), the original cluster before the update phase are shown in dotted lines; (b): overlapping clusters (bottom and upper left) with medium degree (0.71), the merged cluster is nearly the same as the original bottom cluster as the upper left cluster has very low support

defined as follows:

$$S_{cons}(y_A, y_B) = \begin{cases} 1 - \frac{2}{\pi} * \phi & \phi \in [0, \frac{\pi}{2}] \\ \frac{2}{\pi} * (\phi - \frac{\pi}{2}) & \phi \in [\frac{\pi}{2}, \pi] \end{cases} \quad (10)$$

Inspired by Yager's idea of "participatory learning" [18], we propose the following modification of the combination rule (8):

$$\vec{w}_{new} = \vec{w}_A + \alpha \cdot \rho(\vec{w}_A, \vec{w}_B) \cdot (\vec{w}_B - \vec{w}_A), \quad (11)$$

where $\alpha = k_B/(k_A + k_B)$ and $\rho(\vec{w}_A, \vec{w}_B)$ is a measure of consistency of the two rule consequents. This measure can be defined in different ways, e.g., "smoothly" by $\rho(\vec{w}_A, \vec{w}_B) = S_{cons}(y_A, y_B)$ or, more drastically, by

$$\rho(\vec{w}_A, \vec{w}_B) = \begin{cases} 1 & \text{if } S_{cons}(y_A, y_B) \geq OL(A, B) \\ 0 & \text{if } S_{cons}(y_A, y_B) < OL(A, B) \end{cases}.$$

For $\rho = 0$, we obtain $\vec{w}_{new} = \vec{w}_A$, i.e., the consequent of the more relevant rule. For $\rho = 1$, on the other hand, (11) reduces to the original combination rule (8).

3.1.4. Integration in EFS

In case when no cluster is updated but a new one evolved, a rule merging process is obviously not needed. Hence, the integration of the rule merging process is simply accomplished by adding the following steps at the end of each incremental learning cycle:

(1) **If** a new rule was evolved, no action is needed.
(2) **Else** perform the following steps (let $\mathcal{R}$ denote the current set of rules):
(3) Check if overlap (similarity) of moved/updated rule A with any other rule $R \in \mathcal{R} \setminus \{A\}$ calculated by (6) is higher than a pre-defined threshold sim_{thr}
(4) **If** yes
   (a) Perform rule merging of rule A with rule $B = \arg\max_{B \in \mathcal{R} \setminus \{A\}} OL(A, B)$ according to (7).
   (b) Perform merging of corresponding rule consequent functions according to (11).
   (c) Overwrite parameters $(\vec{c}^A, \vec{\sigma}^A)$ of rule A with the parameters of the merged rule $(\vec{c}^{new}, \vec{\sigma}^{new})$.
   (d) Delete rule B.
   (e) Decrease number of rules: $C = |\mathcal{R}| = C - 1$.

3.2. Fuzzy Set Merging in the Partition Space

In the previous section, we proposed a novel rule merging procedure which was acting directly on the complete rule antecedent parts. In this section, we go one step further and introduce an approach for merging two or more single fuzzy sets in the fuzzy partitions of the input variables (without necessarily merging the rules in which these fuzzy sets appear). Redundant fuzzy sets are produced by two or more clusters intersecting in a single dimension, i.e., when being projected on one of the axes of the high-dimensional feature space. A two-dimensional example of such a situation is shown in Fig. 3.

On-line merging of fuzzy sets is necessary in order to assure distinguishability within the fuzzy partitions in each dimension. Distinguishability of the fuzzy sets is a key prerequisite for interpretability of a fuzzy model [19].

3.2.1. Similarity and Merging of Fuzzy Sets

For calculating the similarity of two one-dimensional fuzzy sets, $S_{set}(A, B)$, we adopt

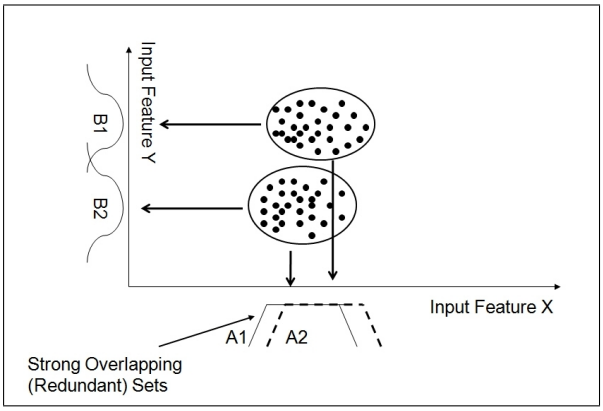


Figure 3: After projection of the clusters to the x-axis, the fuzzy sets are strongly overlapping.

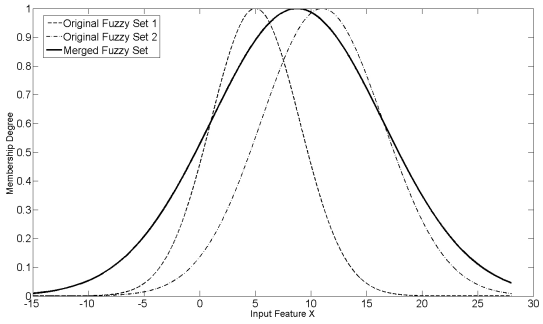


Figure 4: Merging of two Gaussian fuzzy sets (dashed and dotted dashed lines) to a new fuzzy set according to (12)

the same measure as used in (6). Thus, we again use (5) and apply the minimum operator to combine the degree of inclusion of fuzzy set A in B and of fuzzy set B in A. Note that this approach prevents the merging of two fuzzy sets if one of them is contained in the other one, but not vice versa. This is reasonable, since fuzzy sets of that kind usually represent different distributions of data clouds in different regions of the input space. Hence, a merging of such sets would yield an imprecise representation of parts of the data.

If $S_{set}(A, B)$ exceeds a given threshold, the two sets are considered as sufficiently similar (redundant) and are therefore merged. The value of the threshold controls the trade-off between precision and distinguishability. The higher the threshold, the less merging steps will be carried out.

In case of Gaussian membership functions with center μ and spread σ, two fuzzy sets are combined into a new Gaussian kernel with the following parameters:

$$\begin{aligned} \mu_{new} &= (\max(U) + \min(U))/2 \\ \sigma_{new} &= (\max(U) - \min(U))/2 \end{aligned} \tag{12}$$

where $U = \{\mu_A \pm \sigma_A, \mu_B \pm \sigma_B\}$. The idea underlying this definition is to reduce the *approximate* merging of two Gaussian kernels to the *exact* merging of two of their α-cuts, for a specific value of α. Here, we choose $\alpha = \exp(-1/2) \approx 0.6$, which is the membership degree of the inflection points $\mu \pm \sigma$ of a Gaussian kernel with parameters μ and σ.

A merging example is presented in Figure 4.

4. Evaluation

4.1. Experimental Setup

For the purpose of evaluation of the proposed complexity reduction methods, we use the FLEXFIS algorithm as outlined in Section 2. We compare this algorithm with and without redundancy elimination in terms of model complexity (measured by the number of rules) and accuracy (measures by mean absolute error, MAE). Besides, we also measure the additional computational cost caused by the redundancy detection and elimination processes. In the following, we summarize some of the main steps of this algorithm (see [20] for a more detailed description). FLEXFIS uses Gaussian membership functions and the product t-norm in (2). Essentially, this means that every rule corresponds to a Gaussian cluster in the p-dimensional input space (associated with a linear function as consequent part). Before entering the on-line learning model, an initial model of that type if normally constructed off-line, using a small set of input/output examples. Then, in the on-line phase, the model is adapted whenever a new data sample $\vec{x}$ arrives:

- If $\|\vec{x} - \vec{c}_{win}\| \geq \rho$, with $\vec{c}_{win}$ the center coordinates of the nearest cluster and ρ the so-called *vigilance* parameter, then a new rule is created. The center of the corresponding cluster is $\vec{c}_C = \vec{x}$, and its range of influence $\vec{\sigma}_C = 0$.
- Otherwise, the center of the nearest cluster $\vec{c}_{win}$ is updated by moving it towards $\vec{x}$. Moreover, the range of influence is adapted by means of a recursive formula for the variance of a cluster [17].
- The modified/evolved clusters are projected to the axes of the attributes in order to update/evolve the fuzzy partitions in each dimension: the centers and widths of the fuzzy sets are associated with the corresponding center coordinates and the width of the clusters in each dimension (for the widths of the fuzzy sets, a small positive constant ϵ is used as a lower bound in order to avoid numerical instabilities).
- The consequent parts of the rules are adapted. To this end, correction vectors are added to the linear consequent parameters and, likewise, correction matrices to the current inverse Hessian matrices. Then, recursive weighted least squares (for local learning) is performed for all

rules.

The vigilance parameter in FLEXFIS is essential for controlling the trade-off between rule evolution and rule update; thus, it has an important influence on the model complexity. As proposed in [20], we use the default value $0.3\frac{\sqrt{p+1}}{\sqrt{2}}$, with p the dimensionality of the input feature space. The threshold for rule similarity in the feature space as well as for fuzzy set similarity is set to 0.35 in all experiments.

The following data sets were used:

- The jester data set[1] contains the ratings of up to 36 jokes by users on a continuous scale in the range [-10,10]. The goal is to recognize similar rating patterns among different users and predict the rating scores of new users. We used the dense subset of jokes $\{5, 7, 8, 13, 15, 16, 17, 18, 19, 20\}$ as described on the web-site, including the ratings of about 25000 users, in a data streaming context for updating the models on demand based on new user data. The data was randomly split into a training and a test set.
- A data set containing the five most important features for estimating the prices of residential premises. The goal is to predict the house prices for prospective years based on past conditions. The training data set contains values in the period from 1998 to 2004, whereas the test data contains values from the period 2005 to 2006.

The main characteristics of the data sets are summarized in Table 1. In both cases, the training sets were considered as pseudo-streams and used as input of our EFS. The test sets were then used for evaluation, namely to determine the classification accuracy of the respective methods.

4.2. Results

4.2.1. Results on Jester Data

The training samples are processed in four different ways: EFS (without pruning), EFS+ERR (evolving fuzzy system with elimination of redundant rules), EFS+ERF (evolving fuzzy systems with elimination of redundant fuzzy sets) and EFS+ERR+ERF (elimination of redundancies on both levels). For the latter, the combination is done in the following way: for each newly updated rule, it is first checked whether it has become redundant to any other rule — if so, rule merging is performed; then, also all fuzzy sets projected from the updated rule are checked whether they become redundant to any other fuzzy sets — if so, fuzzy set merging is performed and the merged fuzzy partitions stored in a separate (second layer) fuzzy model (for visualization to the user). The update and evolution process continues with the other (first layer) fuzzy model, containing merged clusters, but never merged fuzzy sets. The reason for this is that a fuzzy set merging may cause a cluster dragging and miss-alignment effect, according to a back-projection of the merged sets to the high-dimensional feature space.

Table 2 presents the test results on the jester data when predicting the scores of joke #20 for new users based on past rating patterns. Obviously, our redundancy elimination methods are able to significantly reduce complexity, namely from 467 to only 2 rules, without increasing the mean absolute error (MAE). Indeed, in terms of accuracy, the EFS do even out-perform a batch 1-nearest neighbor classifier used as a baseline. They also come close to Eigentaste (see [21]), a batch method specifically developed for this data set. Computation time is significantly higher when not using any pruning scheme as much more rules are processed during the incremental update phase. Finally, for EFS + ERR, we reach an on-line time of 0.004 seconds for updating the model with a single sample. The accuracy reported for EFS+ERR+ERF belongs to the second layer fuzzy model.

4.2.2. Results on Residential Premise Data

For the residential premise data, the use of an incremental approach can be justified, because the data base is regularly updated with new premises; re-training a model from scratch every time would therefore become very time-consuming.

As test data, we used the prices of two consecutive years 2005-2006, whereas the earlier observations (1998-2004) were used as training data. The results are reported in Table 3. Here, the impact of eliminating redundancies is even more impressive, at the price of a moderate increase in computational cost: The normalized mean absolute error (MAE) decreases from 0.1728 to to 0.1561, probably since complexity reduction also reduces the risk of over-fitting the training data. At the same time, the complexity of the models is reduced, too, from 18 to 7 rules and from 90 to 11 fuzzy sets in total (i.e., from 18 to 2.2 fuzzy sets on average per input dimension).

Figure 5 visualizes the number of rules contained in the model during the on-line update phase — the dotted line represents the original model, the solid line the model obtained through redundancy elimination.

5. Summary and Conclusions

In this paper, we presented methods for eliminating redundancies in evolving rule-based fuzzy systems, thereby reducing the complexity and increasing the transparency of such models. More specifically, we applied corresponding methods on two levels, namely on the level of complete rules (with overlapping antecedent parts) and on the level of the

[1] http://www.ieor.berkeley.edu/ goldberg/jester-data/

Table 1: Data sets and their characteristics

	# Train. Samples	# Test Samples	# Input Var.	Source
Jokes	16654	8329	9	Internet
Premise Prices	2902	1371	5	Historic Data Base of Sales

Table 2: Performance of EFS without and with pruning of redundant rules and fuzzy sets on jester data.

Method	MAE	# of Rules	# of Fuzzy Sets	Comp. Time in sec.
EFS	0.1931	467	4203	817.6
EFS + ERR	0.1958	2	18	77.08
EFS + ERF	0.1926	9	17	1203.2
EFS + ERR + ERF	0.1975	2	11	77.87
1-NN[21]	0.2370	NA	NA	NA
Eigentaste[21]	0.1870	NA	NA	NA

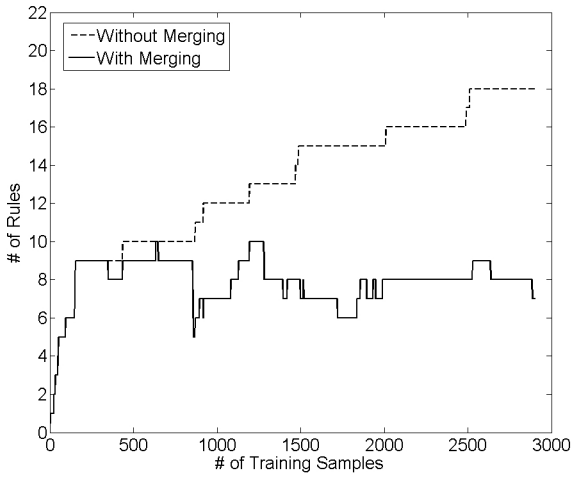


Figure 5: (Evolution progress on the number of rules for NOx data set when using conventional EFS (dotted line) versus EFS with redundancy elimination (solid line).

fuzzy partitions in each single dimension (overlapping fuzzy sets). The degree of overlap of rules and fuzzy sets is quantified by means of a fuzzy inclusion measure. Even though our implementation was evaluated using FLEXFIS approach, the methods are more general and in principle applicable for all evolving fuzzy systems of the Takagi-Sugeno type. Moreover, they can be used in connection with arbitrary fuzzy sets and t-norm operators in the rule antecedent parts. Evaluations on high-dimensional data sets show that complexity reduction does not only produce more compact models, but may also yield a gain in accuracy.

Future work includes the application of the proposed approaches onto dynamically evolving cluster models. There, it would be interesting to see the impact of rule merging for cluster which are moving together onto the quality of evolved cluster partitions (e.g. measured in terms of cluster validation indices).

Acknowledgements

This work was funded by the Austrian fund for promoting scientific research (FWF, contract number I328-N23, acronym IREFS) and the German Research Foundation (DFG). It reflects only the authors' views. The residential premise data set was provided by Bogdan and Krysztof Trawinski.

References

[1] N. Kasabov, *Evolving Connectionist Systems: The Knowledge Engineering Approach - Second Edition.* London: Springer Verlag, 2007.

[2] S. Pang, S. Ozawa, and N. Kasabov, "Incremental linear discriminant analysis for classification of data streams," *IEEE Transactions on Systems, Men and Cybernetics - part B: Cybernetics*, vol. 35, no. 5, pp. 905–914, 2005.

[3] E. Lughofer and P. Angelov, "Handling drifts and shifts in on-line data streams with evolving fuzzy systems," *Applied Soft Computing*, vol. 11, no. 2, pp. 2057–2068, 2011.

[4] E. Lughofer, "Process safety enhancements for data-driven evolving fuzzy models," in *Proceedings of 2nd Symposium on Evolving Fuzzy Systems*, Lake District, UK, 2006, pp. 42–48.

[5] P. Angelov, "Evolving takagi-sugeno fuzzy systems from streaming data, eTS+," in *Evolving Intelligent Systems: Methodology and Applications*, P. Angelov, D. Filev, and N. Kasabov, Eds. New York: John Wiley & Sons, 2010, pp. 21–50.

[6] E. Lughofer, *Evolving Fuzzy Systems — Methodologies, Advanced Concepts and Applications.* Berlin Heidelberg: Springer, 2011, iSBN: 978-3-642-18086-6.

Table 3: Performance of EFS without and with pruning of redundant rules and fuzzy sets on residential premise data.

Method	MAE	# rules	# fuzzy sets	comp. time in sec.
EFS	0.1728	18	90	5.74
EFS + ERR	0.1562	7	35	14.33
EFS + ERF	0.1695	10	15	8.09
EFS + ERR + ERF	0.1561	7	11	14.46

[7] J. Gama, *Knowledge Discovery from Data Streams*. Boca Raton, Florida: Chapman & Hall/CRC, 2010.

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

[9] E. Lima, M. Hell, R. Ballini, and F. Gomide, "Evolving fuzzy modeling using participatory learning," in *Evolving Intelligent Systems: Methodology and Applications*, P. Angelov, D. Filev, and N. Kasabov, Eds. New York: John Wiley & Sons, 2010, pp. 67–86.

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

[11] J. Ramos and A. Dourado, "Pruning for interpretability of large spanned eTS," in *Proceedings of the 2006 International Symposium on Evolving Fuzzy Systems*, Ambleside, UK, 2006, pp. 55–60.

[12] P. Angelov and D. Filev, "Simpl_eTS: A simplified method for learning evolving Takagi-Sugeno fuzzy models," in *Proceedings of FUZZ-IEEE 2005*, Reno, Nevada, U.S.A., 2005, pp. 1068–1073.

[13] E. Lughofer, E. Hüllermeier, and E. Klement, "Improving the interpretability of data-driven evolving fuzzy systems," in *Proceedings of EUSFLAT 2005*, Barcelona, Spain, 2005, pp. 28–33.

[14] E. Klement, R. Mesiar, and E. Pap, *Triangular Norms*. Dordrecht Norwell New York London: Kluwer Academic Publishers, 2000.

[15] E. Lughofer, *Evolving Fuzzy Models — Incremental Learning, Interpretability and Stability Issues, Applications*. Saarbrücken: VDM Verlag Dr. Müller, 2008.

[16] L.Ros, A.Sabater, and F.Thomas, "An ellipsoidal calculus based on propagation and fusion," *IEEE Transactions on Systems, Man and Cybernetics - Part B: Cybernetics*, vol. 32, no. 4, pp. 430–442, 2002.

[17] S. Qin, W. Li, and H. Yue, "Recursive PCA for adaptive process monitoring," *Journal of Process Control*, vol. 10, pp. 471–486, 2000.

[18] R. R. Yager, "A model of participatory learning," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 20, pp. 1229–1234, 1990.

[19] S. Zhou and J. Gan, "Low-level interpretability and high-level interpretability: a unified view of data-driven interpretable fuzzy systems modelling," *Fuzzy Seta and Systems*, vol. 159, no. 23, pp. 3091–3131, 2008.

[20] E. Lughofer, "FLEXFIS: A robust incremental learning approach for evolving TS fuzzy models," *IEEE Transactions on Fuzzy Systems*, vol. 16, no. 6, pp. 1393–1410, 2008.

[21] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins, "Eigentaste: A constant time collaborative filtering algorithm," *Information Retrieval*, vol. 4, no. 2, pp. 133–151, 2001.