Reducing the Complexity in Genetic Learning of Accurate Regression TSK Rule-Based Systems

Ismael Rodríguez-Fdez, Manuel Mucientes and Alberto Bugarín
Centro de Investigación en Tecnoloxías da Información (CITIUS)
Universidade de Santiago de Compostela
{ismael.rodriguez, manuel.mucientes, alberto.bugarin.diz}@usc.es

***Abstract*—In many real problems the regression models have to be accurate but, also, interpretable in order to provide qualitative understanding of the system. In this realm, the use of fuzzy rule base systems, particularly TSK, is widely extended. TSK rules combine the interpretability and expressiveness of rules with the ability of fuzzy logic for representing uncertainty, and the precision of the polynomials in the consequents. In this paper we present a new genetic fuzzy system to automatically learn accurate and simple linguistic TSK fuzzy rule bases that accurately model regression problems. In order to reduce the complexity of the learned models while keeping a high accuracy, we propose a Genetic Fuzzy System which consists of three stages: instance selection, multi-granularity fuzzy discretization of the input variables, and the evolutionary learning of the rule base using Elastic Net regularization. This proposal was validated using 28 real-world datasets and compared with three state of the art genetic fuzzy systems. Results show that our approach obtains the simplest models while achieving a similar accuracy to the best approximative models.**

***Index Terms*—Genetic Fuzzy Systems, regression, instances selection, multi-granularity fuzzy discretization**

I. Introduction

The main objective of a machine learning algorithm is to obtain models that accurately predict the expected output using the provided inputs. In addition, the application of these models generally requires interpretability [1], since it is desirable to have information that facilitates the qualitative understanding of the model. Moreover, when the complexity of the learned models is high, although the training error decreases, the models overfits the training data and, thus, the test error increases. On the other hand, when the model is too simple, it can underfit the data and produce a high training and test error.

For regression problems, the use of fuzzy rule base systems is very extended. It usually combines the interpretability and expressiveness of the rules with the ability of fuzzy logic for representing uncertainty. One of the most widely used learning algorithms for automatic modeling of fuzzy rule bases are Genetic Fuzzy Systems (GFSs) [2], i.e., the combination of evolutionary algorithms and fuzzy logic. Evolutionary algorithms are appropriate for learning fuzzy rules due to their flexibility —that allows them to codify any part of the fuzzy rule base system—, and due to their capability to manage the balance between accuracy and complexity of the model in an effective way.

Fuzzy sets improve modeling tasks in regression problems, particularly through the use of Takagi Sugeno Kang (TSK) systems [3]. In particular, recent developments using multi-objective evolutionary fuzzy systems can be found in [4], [5], where both Mamdani and TSK systems were proposed to solve large-scale regression problems. Moreover, in [6] an adaptive fuzzy inference system was proposed to cope with high dimensional problems.

The interpretability of GFSs for regression has been mostly addressed through the control of the complexity of the rule base and, particularly, taking into account the number of rules and/or the number of labels through a multi-objective approach [7]. More recently, the use of instance selection techniques has received more attention in both classification [8], [9] and regression [10] problems. This approach faces two problems at once: decreases the complexity for large-scale problems and reduces the overfitting, as the rules can be generated with a part of the training data and the error of the rule base can be estimated with another part (or the whole) training set.

Furthermore, the complexity of fuzzy rule base models is highly influenced by the number of labels (granularity) used for each variable. When no expert knowledge is available to determine the fuzzy labels, two different approaches can be applied: uniform discretization combined with lateral displacements [11], or non-uniform discretization [12]. Moreover, the data base can be of two types: (i) linguistic, in which all rules share the same fuzzy partition for each variable; (ii) and approximative, which uses a different definition of the fuzzy labels for each rule in the rule base. The former implies more interpretability and simpler models while the latter usually obtains more accurate solutions. Recently, [13], [14] have applied non-uniform discretization techniques to classification problems.

Finally, the use of TSK fuzzy rule bases implies another complexity dimension: the polynomial in the consequent — usually with degree 1 or 0. Prediction accuracy can be achieved learning the consequent through the least squares method. However, that choice often provides a low bias but a large variance. This problem can be solved by shrinking (ℓ_2, also known as Ridge regularization) or setting some coefficients to zero (ℓ_1, also called Lasso regularization). Moreover, a combination of both regularizations, called Elastic Net [15] can be used.

In this paper we propose the new GFS algorithm L-TSK

(Linguistic TSK), for obtaining accurate and simple linguistic TSK (type-1) fuzzy rule base models to solve regression problems. The main contributions of this work are: i) a new instance selection method for regression, ii) a novel multi-granularity fuzzy discretization of the input variables, in order to obtain non-uniform fuzzy partitions with different degrees of granularity, iii) an evolutionary algorithm that uses a fast and scalable method with Elastic Net regularization to generate accurate and simple TSK (type-1) fuzzy rules.

This work is structured as follows: Sec. II describes the different stages of the GFS: the instance selection method, the discretization approach, and the evolutionary algorithm. Sec. III shows the results of the approach in 28 regression problems, and the comparison with other proposals through statistical tests. Finally, the conclusions are presented in Sec. IV.

II. Proposed Method

This section presents the three main components of our method: a two-stage preprocessing —composed of the instance selection and multi-granularity fuzzy discretization modules— , and a genetic algorithm (Fig. 1). Both preprocessing techniques are executed in order to improve the simplicity of the fuzzy rule bases obtained by the evolutionary algorithm. On one hand, the instance selection reduces the variance of the models focusing the generated rules on the representative examples. On the other hand, the multi-granularity fuzzy discretization decreases the complexity of the fuzzy partitions and, therefore, it is not necessary to establish an upper bound in the number of rules in the evolutionary stage.

A. Instance Selection for Regression

The instance selection method for regression is an improvement of CCISR (Class Conditional Instance Selection for Regression) [10]. The main differences with CCISR are:

- The output variable is discretized in order to simplify the generation of the different graphs of the process.
- The error measure is based on the $1-nearest\ neighbor$ ($1NN$) for regression, thus reducing the complexity of the calculations.

The instance selection method is based on a relation called class conditional nearest neighbor (*ccnn*) [16], defined on pairs of points from a labeled training set as follows: for a given class c, $ccnn$ associates to instance a its $nearest\ neighbor$ computed among only those instances (excluded a) in class c. Thus, this relation describes proximity information conditioned to a class label.

In regression problems, the outputs are real values instead of labels and, therefore, they must be discretized in order to use the *ccnn* relation. Traditionally, an unsupervised discretization process needs the definition of the number of intervals or their shape [17]. In our approach, the shape of the intervals is guided by the output density, i.e., the split points are selected in the zones where the density of the output is minimum.

We use Kernel Density Estimation (KDE) with a gaussian kernel in order to estimate the probability density function of the output variable (y) in a non-parametric way. In order to select the appropriate kernel bandwidth, Scott's rule is applied. [18]. Once the probability density function is obtained, the local minimum determines the split points, and, therefore, which labels/classes are used for the *ccnn* relation.

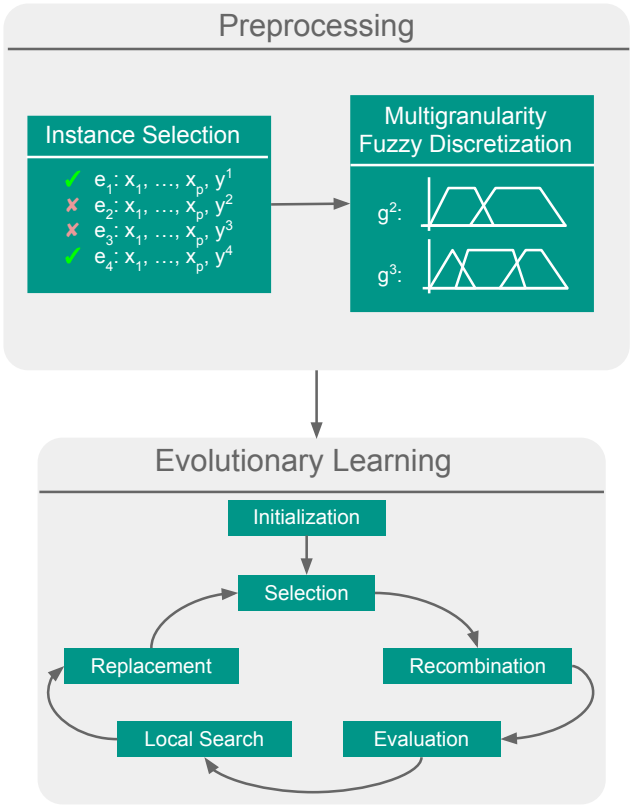

Figure 1. The Genetic Fuzzy System with the three stages.

Two different graphs can be constructed using this relation:

- Within-class directed graph (G_{wc}): consists in a graph where each instance has an edge pointing to the nearest instance of the same class.
- Between-class directed graph (G_{bc}): is a graph where each instance has an edge pointing to the nearest instance of any different class.

These graphs are used to define an instance scoring function by means of a directed information-theoretic measure (the K-divergence) applied to the in-degree distributions of these graphs. The scoring function (named `Score`) is used to develop an effective large margin instance selection method, called Class Conditional selection (Fig. 2).

The instance selection algorithm starts from a set of training examples:

$$E = \{e^1, e^2, \ldots, e^n\} \tag{1}$$

where n is the number of examples. Also, it uses the *leave-one-out* mean squared error (MSE) with *1NN* (this error is called ϵ) in order to estimate the information loss.

First, an initial core of instances from E is selected, sorted by `Score`. The size of this initial set is:

$$k_0 = max\left(c, \left\lceil \frac{\epsilon^E \cdot |E|}{max(y) - min(y)} \right\rceil\right) \quad (2)$$

where c is the number of classes obtained from KDE and ϵ^E is the error using the set of examples in E. This choice is motivated because (i) there is at least one example for each class, and (ii) the error in the second part can be interpreted as the miss-classification probability. After this, the instance selection method iteratively selects instances and adds them to set S, choosing in first place those with the highest score. The process terminates when the empirical error (ϵ^S) increases.

1: $\{e^1, \ldots, e^n\} = E$ sorted in decreasing order of `Score`
2: $S = \{e^1, \ldots, e^{k_0}\}$
3: $go\_on = true$
4: $ub = n - |\{e^l \ / \ \texttt{Score}(e^l) <= 0\}|$
5: $l = k_0 + 1$
6: **while** $l < ub \wedge go\_on$ **do**
7: $\quad Temp = S \cup \{e^l\}$
8: $\quad$ **if** $\epsilon^S \leq \epsilon^E$ **then**
9: $\quad\quad go\_on = false$
10: $\quad$ **if** $\epsilon^{Temp} < \epsilon^S \wedge go\_on$ **then**
11: $\quad\quad S = Temp$
12: $\quad\quad l = l + 1$
13: $\quad$ **else**
14: $\quad\quad go\_on = false$
15: return S

Figure 2. Pseudocode of Class Conditional selection [16].

In order to further improve the number of selected instances, the method uses the Thin-out post-processing (Fig. 3). This algorithm selects points close to the decision boundary of the *1NN* rule. This is achieved by selecting instances having positive in-degree in the between-class graph set S (G^S_{bc}) and storing them in S_f. Also, S_1 is the subset of examples that are in S but not in S_f. Then an iterative process is done as follows: points having positive in-degree in the $G^{S_1}_{bc}$ are added to S_f if they were not "isolated" in the previous iteration, that is, if their in-degree was not zero (line 6). This iterative process terminates when the empirical error increases (line 7).

B. Multi-granularity Fuzzy Discretization for Regression

The objective of fuzzy discretization is to automatically obtain a good set of fuzzy labels for a variable. Furthermore, multi-granularity refers to the definition of a different number of fuzzy labels for each granularity. Specifically, a granularity g^i_{var} divides the variable var in i fuzzy labels, i.e., $g^i_{var} = \{A^{i,1}_{var}, \ldots, A^{i,i}_{var}\}$. In order to preserve interpretability between contiguous granularities, we follow a hierarchical top-down approach for the multi-granularity discretization: the linguistic fuzzy labels of a granularity are equal to the labels of

1: $S_f = \{e^l \in S$ with in-degree in $G^S_{bc} > 0\}$
2: $S_{prev} = S$
3: $S_1 = S \setminus S_f$
4: $go\_on = true$
5: **while** go_on **do**
6: $\quad S_t = \{e \in S_1$ with in-degree in $G^{S_1}_{bc} > 0$ and with in-degree in $G^{S_{prev}}_{bc} > 0\}$
7: $\quad go\_on = \epsilon^{S_f \cup S_t} < \epsilon^{S_f}$
8: $\quad$ **if** go_on **then**
9: $\quad\quad S_f = S_f \cup S_t$
10: $\quad\quad S_{prev} = S_1$
11: $\quad\quad S_1 = S \setminus S_f$
12: return S_f

Figure 3. Pseudocode of Thin-out selection [16].

the previous granularity, except for one of the previous labels which is replaced by two new fuzzy linguistic labels.

In regression problems (TSK type-1 in our case), the discretization process must search for the split point that minimizes the error when a linear model is applied to each of the resulting intervals. Hence, in a top-down approach only a new split point is added at each step, obtaining two new intervals. The generation of the fuzzy linguistic labels can be divided into two stages. First, the variable X must be discretized to obtain a set of split points C^g for each granularity g. Then, given the split points, the fuzzy labels can be defined for each granularity.

In order to select the maximum number of split points for a variable, we have used the well-known Bayesian Information Criterion (BIC). This measure can be separated into two parts: the error and the complexity of the model. In this case, the error is obtained from the summation of the MSE of a least squares fitted model for each interval of the discretization. On the other hand, the complexity of the model is determined by the number of parameters, in this case the number of inner splits and the parameters fitted by each regression.

The pseudocode of the discretization method for a variable is shown in Fig. 4. First, the split points for granularity 1 are defined using the domain limits (line 2). The BIC measure for this first granularity is calculated (line 3) using MSE, a function that gets a set of examples X, learns a linear regression model using least squares and, finally, calculates the mean squared error of the model. In this case, the number of parameters is two, corresponding to the coefficients of the linear model. After that, an iterative process is executed: at each step, the split points of a new granularity are defined adding a new split point to the previous granularity (lines 4-15).

In order to obtain the split point for the new granularity, first, the best split point (c_i) for each interval between the split points of the previous granularity ($[C^g_i, G^g_{i+1}$) is obtained using the golden section search method [19]. This method searches for the value that minimizes the function `LinearError` (Fig.

1: $g = 1$
2: $C^g = \{min(X), max(X)\}$
3: $BIC^g = |X| \cdot log(MSE(X)) + 2 \cdot log(|X|)$
4: **repeat**
5: c_i =GOLDENSECTIONSEARCH(C_i^g, C_{i+1}^g, LINEARERROR($\{x \in X : C_i^g < x < C_{i+1}^g\}$)) $\forall i = 0...|C^g| - 1$
6: $i_{min} = \text{argmin}_i$ LINEARERROR($\{x \in X : C_i^g < x < C_{i+1}^g\}$, c_i)
7: $C^{g+1} = C^g \cup \{c_{i_{min}}\}$
8: $g = g + 1$
9: $BIC^g = |X| \cdot log(\sum_{i=0}^{|C^g|-1} MSE(\{x \in X : C_i^g < x < C_{i+1}^g\}) + (|C^g| - 2) \cdot 2 \cdot log(|X|)$
10: **if** $BIC^g < BIC^{min}$ **then**
11: $it = 0$
12: $min = g$
13: **else**
14: $it = it + 1$
15: **until** $min > \sqrt{|X|/it}$
16: **return** $C^1, \ldots, C^{min}$

Figure 4. Pseudocode of the discretization method.

5) in an interval. `LinearError` gets a set of examples X and a split point c and calculates the total squared error of X, which is calculated with the corresponding linear regression models at each side of the split point. Then, the split point that minimizes the `LinearError` is selected and added to the new granularity split points (lines 6-7), and the BIC measure is calculated (line 9). The number of parameters used for the BIC measure is 2 (coefficients of the linear regression) for each interval. The number of intervals is calculated as $|C^g|-2$, where 2 is subtracted to disregard the split points at the end of the variable X domain.

1: **function** LINEARERROR(X, c)
2: $X_l = \{x \in X : x < c\}$
3: $X_r = \{x \in X : x > c\}$
4: **return** $SE(X_l) \cdot \frac{|X_l|}{|X|} + SE(X_r) \cdot \frac{|X_r|}{|X|}$

Figure 5. Pseudocode of the function to be minimized by the discretization method.

Finally, when the granularity with minimum BIC is greater than $\sqrt{|X|/it}$, where it is the number of iterations without improvement in the BIC value, the algorithm stops (lines 10-15). This criteria ensures that at the beginning of the discretization process —the granularity is low—, the BIC may worsen for more iterations, while with larger granularities, the algorithm is more strict in the stopping criterion.

After obtaining the discretization of the variable for each granularity, the method proposed in [12] is applied for each C^g in order to obtain the multi-granularity fuzzy partitions. This method uses a fuzziness parameter that indicates how fuzzy are the linguistic labels. A fuzziness of 0 indicates crisp intervals, while a fuzziness of 1 indicates the selection of a fuzzy set with the smallest kernel —set of points with membership equal to 1.

C. Evolutionary Algorithm

The evolutionary algorithm is used in order to learn a linguistic TSK fuzzy rule base model. The integration of the evolutionary algorithm with the preprocessing stage is as follows:

- First, the instance selection process is executed over the training examples E_{tra} in order to obtain a subset of representative examples E_S
- Then, the multi-granularity fuzzy discretization process obtains the fuzzy partitions for each input variable.
- Finally, the evolutionary algorithm searches for the best combination of granularities and generates the entire linguistic TSK fuzzy rule base, using both E_{tra} and E_S.

In what follows, we describe in detail the different components of the evolutionary algorithm.

1) Chromosome Codification: A double-coding scheme is used: a vector of granularities for each variable (C_1) and a vector of real numbers that represent lateral displacements (C_2). The second part of the chromosome is based on the 2-tuple representation of the labels [20]. This approach applies a displacement of a linguistic term within the $[-0.5, 0.5)$ interval that expresses the movement of a label between its two adjacent labels. However, since the keypoints for the fuzzy partitions are the split points obtained from the multi-granularity fuzzy discretization (Sec. II-B), the lateral displacements are applied to these split points. Thus, the length of C_2 depends on the granularity for each input variable: $|C_2| = \sum (g - 1), \forall g \in C_1$.

2) Initialization: The initial pool of individuals is generated by a combination of two initialization procedures. A half of the individuals are generated with the same random granularity for each variable, while the other half is created with a different random granularity for each variable. In either case, when the product of the granularities indicated in C_1 (i.e., the maximum number of rules that can be obtained) is greater than the product of the number of input variables times the hightest maximum granularity of the variables, then a variable is randomly selected and its granularity is set to 0 until the previous condition is no longer satisfied. This is done in order to avoid too complex solutions in the initialization stage —during the evolutionary learning this upper bound to the number of rules does not apply. The lateral displacements are initialized to 0 in all cases.

3) Fast Rule Base Generation: An ad-hoc method is used to construct the rule base from the data base codified in the chromosome, i.e. the fuzzy partitions indicated in C_1 after applying the displacement in C_2. The Wang & Mendel algorithm [21] is used to create the antecedent part of the rule base for each individual. The method is quick and simple, and obtains a representative rule base given the definition of the data base and a set of examples.

The consequent part of the rules is learned using the Elastic Net method [15] in order to obtain the coefficients of the degree 1 polynomial for each rule. Thus, overfitting is avoided using the double regularization (ℓ_1 and ℓ_2) of Elastic Net. In order to obtain a solution using the Elastic Net method, the Stochastic Gradient Descent (SGD) optimization technique is used [22], [23]. This approach provides a fast and scalable method for obtaining regression models.

Only those examples in E_s are used to obtain the rule base from the codified chromosome. In this manner, those examples that are not representative are not taken into account for the generation of the rules. Thus, the method avoids the creation of too specific rules, and reduces the time needed to create the rule base.

4) Evaluation: The fitness function is based on the estimation of the error of the generated rule base:

$$fitness = MSE(E_{tra}) \tag{3}$$

where E_{tra} is the full training dataset. Using all the examples for evaluation can be seen, in some way, as a validation process, as the rule base was constructed with a subset of them (E_S).

5) Selection and Replacement: The selection is performed by a binary tournament. On the other hand, the replacement method joins the previous and current populations, and selects the N best individuals as the new population.

6) Recombination: Two crossover operations are defined: one-point crossover for exchanging the C_1 parts (it also exchanges the corresponding C_2 genes) and, when the C_1 parts are equal , the parent-centric BLX (PCBLX) [24] is used to crossover the C_2 part. In order to prevent the crossover of too similar individuals, an incest prevention is implemented. When the euclidean distance of the lateral displacements is less than a particular threshold L, the individuals are not crossed.

The mutation (applied with probability p_{mut}) consists in the application of one of two possible operations with equal probability to a randomly selected gene of the C_1 part: i) decreasing the granularity by 1 or ii) increasing the granularity to a more specific granularity (all the granularities have the same chance). In order to calculate the new lateral displacements in the corresponding C_2 part, the displacements of the previous granularity are taken into account. The displacement associated with a particular split point is calculated adding the displacements of the nearest split points of the previous granularity (before mutation) weighted by the distance between the split points.

7) Local Search: After the replacement, all the new individuals (i.e., the C_1 part of the chromosome was not tried before) are used in a local search process. This stage generates n_{ls} new C_1 parts with equal or less granularity —with equal probability— than the selected individual. Then, the C_2 part is generated randomly with a uniform distribution in the $[-0.5, 0.5)$ interval. The new chromosomes are decoded and evaluated and, if any solution obtains better fitness, then it replaces the original individual.

Table I
THE 28 DATASETS OF THE EXPERIMENTAL STUDY.

Problem	Abbr.	Variables	Cases
Electrical Length	ELE1	2	495
Plastic Strength	PLA	2	1650
Quake	QUA	3	2178
Electrical Maintenance	ELE2	4	1056
Friedman	FRIE	5	1200
Auto MPG6	MPG6	5	398
Delta Ailerons	DELAIL	5	7129
Daily Electricity Energy	DEE	6	365
Delta Elevators	DELELV	6	9517
Analcat	ANA	7	4052
Auto MPG8	MPG8	7	398
Abalone	ABA	8	4177
Concrete Compressive Strength	CON	8	1030
Stock prices	STP	9	950
Weather Ankara	WAN	9	1609
Weather Izmir	WIZ	9	1461
Forest Fires	FOR	12	517
Mortgage	MOR	15	1049
Treasury	TRE	15	1049
Baseball	BAS	16	337
California Housing	CAL	8	20640
MV Artificial Domain	MV	10	40768
House-16H	HOU	16	22784
Elevators	ELV	18	16559
Computer Activity	CA	21	8192
Pole Telecommunications	POLE	26	14998
Pumadyn	PUM	32	8192
Ailerons	AIL	40	13750

8) Restart and Stop Criteria: The restart mechanism uses the incest prevention threshold L. First, L is initialized as the maximum length of the C_2 part, i.e. the product of the number of input variables times the biggest maximum granularity of the variables, divided by 4. This implies that the incest prevention allows crossovers between individuals that have a distance higher than a quarter of the maximum euclidean distance. Then, for each iteration, L is decreased in different ways:

- L is decreased by 0.2 in all the iterations, in order to increase convergence.
- If there is no new individual in the population, then L is decreased by 0.1.
- If the best individual does not change, L is also decreased by 0.1.

Finally, when L reaches 0, the population is restarted, and L is reinitialized. Only the best individual so far is kept, and in order to complete the population, the local search process is executed using the chromosome of the best individual. When the restart criterion is fulfilled twice, the algorithm stops, i.e., one single restart is executed.

III. RESULTS

In order to analyze the performance of the proposal, we have used 28 real-world regression problems with different complexity. These problems were obtained from the KEEL project [25]. Table I shows the characteristics of the datasets.

A. Experimental Setup

In order to assess L-TSK performance, we compare its performance to other three genetic approaches which are among the most accurate genetic fuzzy systems for regression in the literature:

- $FS_{MOGFS}{}^{e}$+TUN^{e} [4]: a multi-objective evolutionary algorithm that learns approximative Mamdani fuzzy rule bases. This algorithm learns the granularities from uniform multi-granularity fuzzy partitions (up to granularity 7) and lateral displacement of the labels. It includes a post-processing algorithm for tuning parameters of the membership functions and for rule selection.
- L-METSK-HD^{e} [5]: a multi-objective evolutionary algorithm that learns linguistic type-0 TSK fuzzy rule bases. The algorithm learns the granularities from uniform multi-granularity fuzzy partitions (up to granularity 7).
- A-METSK-HD^{e} [5]: a multi-objective evolutionary algorithm that learns approximative type-1 TSK fuzzy rule bases. The algorithm starts with the solution obtained on the first stage and applies a tuning of the membership functions, rule selection and a Kalman-based calculation of the consequents of the rules.

Regarding the parameters used for the approach presented in this paper, the proposal was designed to keep the number of parameters as low as possible. For the instance selection technique, no parameters are needed. In the multi-granularity fuzzy discretization, a $1E-5$ precision value was used for the golden section search to stop. The fuzziness parameter used for the generation of the fuzzy intervals from the split points was 1, i.e., the highest fuzziness value. For the evolutionary algorithm, the values of the parameters were: population size of 61, a maximum number of evaluations of $50,000$, 0.2 for the mutation probability p_{mut}, and 5 for the n_{ls}. For the generation of the TSK fuzzy rule bases, the weight of the tradeoff between ℓ_1 and ℓ_2 regularizations on the Elastic Net was set to 0.95 and the maximum number of iterations of the SGD was 100.

A 5-fold cross validation was used in all the experiments. Moreover, 6 trials (with different seeds for the random number generation) were executed for each 5-fold cross validation. Thus, a total of 30 runs were obtained for each dataset. The results shown in the next section are the mean value of the 30 runs.

B. Statistical Analysis

Table II shows the average results of the approach in this paper (L-TSK) and the three algorithms selected for comparison. Two different results are shown for each algorithm and dataset: the number of rules of the obtained rule base, and the test error. These indicators allow to compare both the accuracy and the complexity of the learned models. In some papers, the number of labels is also taken into account to measure the simplicity of the rule bases, but since our proposal generates all the possible rules of a data base, the number of labels is implicit in the number of rules. Moreover, the values with the best accuracy —lowest error— and best number of rules are marked in bold face.

It can be seen that the number of rules of our approach is the lowest in the majority of the datasets. It should be noted that the number of rules in the large scale problems (the last 8 problems) is also low despite the high number of examples. Only in 5 problems the $FS_{MOGFS}{}^{e}$+TUN^{e} Mamdani proposal produces a lower number of rules. In the case of accuracy, in 16 of the 28 problems our approach achieves the best results. In the other 12 datasets, the best results are for $FS_{MOGFS}{}^{e}$+TUN^{e} (best in 3 problems) and A-METSK-HD (best in 9 problems).

In order to analyze the statistical significance of these results the STAC platform [26] was used to apply the statistical tests. A Friedman test was used for both the number of rules and the test error in order to get a ranking of the algorithms and check if the differences between them are statistically significant.

Table III shows the ranking for the test error, with the p-value of the test. Our proposal gets the lowest ranking, i.e., has the best results in accuracy among all the algorithms. Then, the next algorithms in the ranking are the approximative approaches, due to its fine tuning of the rules, followed by the linguistic approach L-METSK-HD. In order to compare whether the difference between our proposal and the approximative ones is significant, a Holm post-hoc procedure was performed (Table IV). The differences are statistically significant against the $FS_{MOGFS}{}^{e}$+TUN^{e} Mamdani proposal (p-value below 0.01) and L-METSK-HD, but not with the A-METSK-HD TSK-based approach. However, even with a linguistic representation of the rules, our approach obtains greater accuracy than the approximative approaches, while getting simpler models.

In order to compare the complexity of the models obtained for each algorithm, the same Friedman test was performed to the number of rules in table II (Table V). Once again, our proposal has the lowest ranking. The next algorithm in the ranking is the $FS_{MOGFS}{}^{e}$+TUN^{e} Mamdani approach, followed by the METSK-HD approaches. In order to assess whether the difference among the proposals is significant, a Holm post-hoc procedure was also performed (Table VI). The differences are statistically significant (p-values below a significance level of 0.1). This shows that our approach obtains the simpler models among all the methods.

Although each of the steps performed in the algorithm increase the computational complexity of our approach, they contribute to focus the search on the simplest models. The time consumed by L-TSK is in the same order of magnitude as A-METSK-HD using a processor with similar characteristics. Our method obtains solutions in the range between 1 to 23 minutes for datasets 1-20 (the most simple ones) and solutions in the range from 1 hour to 30 hours for datasets 21-28 (the most complex ones).

Table II
AVERAGE RESULTS FOR THE DIFFERENT ALGORITHMS. THE TEST ERRORS IN THIS TABLE SHOULD BE MULTIPLIED BY $10^5, 10^{-8}, 10^{-6}, 10^9, 10^8, 10^{-6}, 10^{-4}, 10^{-8}$ IN THE CASE OF ELE1, DELAIL, DELELV, CAL, HOU, ELV, PUM, AIL RESPECTIVELY.

algorithms	L-TSK		FS_{MOGFS}^{e}+TUN^{e}		L-METSK-HD		A-METSK-HD	
	# Rules	Test Error	# Rules	Test Error	# Rules	Test Error	# Rules	Test Error
ELE1	**2**	2.1622	8.1	**1.954**	15	1.925	11.4	2.022
PLA	**1**	1.1792	18.6	1.194	23	1.218	19.2	**1.136**
QUA	7	0.0181	**3.2**	**0.0178**	35.9	0.0185	18.3	0.0181
ELE2	**4.5**	7265	8	10548	59	20095	36.9	**3192**
FRIE	**8**	**0.7276**	22	3.138	95.1	3.084	66	1.888
MPG6	**9.9**	**3.6438**	20	4.562	99.6	4.469	53.6	4.478
DELAIL	**4.8**	1.4646	6.2	1.528	98.3	1.621	36.8	**1.402**
DEE	**5.1**	**0.0805**	18.3	0.093	96.4	0.095	50.6	0.103
DELELV	**6**	1.0463	7.9	1.086	91	1.119	39.1	**1.031**
ANA	**3.2**	0.0091	10	**0.003**	48.9	0.006	33.3	0.004
MPG8	**7.9**	**3.9065**	23	4.747	98.7	5.61	64.2	5.391
ABA	**5.3**	**2.3482**	8	2.509	42.4	2.581	23.1	2.392
CON	**7**	**21.066**	15.4	32.977	96.5	38.394	53.7	23.885
STP	65.2	**0.3403**	**23**	0.912	100	0.78	66.4	0.387
WAN	**5.4**	**0.9408**	8	1.635	91.1	1.773	48	1.189
WIZ	13.1	**0.6749**	**10**	1.011	55.4	1.296	29.1	0.944
FOR	**5.3**3	**2118**	10	2628	93.7	4633	40.6	5587
MOR	**6.9**	**0.0072**	7	0.019	40.9	0.028	27.2	0.013
TRE	**6.9**	**0.0282**	9	0.044	42.8	0.052	28.1	0.038
BAS	**5.6**	**259452**	17	261322	95.7	320133	59.8	368820
CAL	32	2.4034	**8.4**	2.95	99.8	2.638	55.8	**1.71**
MV	**7.4**	0.0934	14	0.158	76.4	0.244	56.5	**0.061**
HOU	**9.2**	14.4553	11.7	9.4	68.9	10.368	30.5	**8.64**
ELE	**6.4**	**2.9132**	8	9	76.4	8.9	34.9	7.02
CA	**9.3**	**4.6511**	14	5.216	71.3	5.88	32.9	4.949
POLE	40.1	136.7707	**13.1**	102.816	100	150.673	46.3	**61.018**
PUM	**8**	0.5403	17.6	0.292	87.5	0.594	63.3	**0.287**
AIL	**13.4**	**1.402**	15	2	99.1	1.822	48.4	1.51

Table III
FRIEDMAN TEST RANKING RESULTS FOR THE TEST ERROR IN TABLE II.

Algorithm	Ranking
L-TSK	1.73
A-METSK-HD	2.018
FS_{MOGFS}^{e}+TUN^{e}	2.786
L-METSK-HD	3.464
p-value	$< 1E-5$

Table IV
HOLM POST-HOC ADJUSTED P-VALUES FOR THE TEST ERROR RANKING IN TABLE III.

Comparison	Adjusted p-value
L-TSK vs L-METSK-HD	$< 1E-5$
L-TSK vs FS_{MOGFS}^{e}+TUN^{e}	0.004
L-TSK vs A-METSK-HD	0.408

Table V
FRIEDMAN TEST RANKING RESULTS FOR THE NUMBER OF RULES IN TABLE II.

Algorithm	Ranking
L-TSK	1.178
FS_{MOGFS}^{e}+TUN^{e}	1.821
A-METSK-HD	3
L-METSK-HD	4
p-value	$< 1E-5$

Table VI
HOLM POST-HOC ADJUSTED P-VALUES FOR THE NUMBER OF RULES RANKING IN TABLE V.

Comparison	Adjusted p-value
L-TSK vs L-METSK-HD	$< 1E-5$
L-TSK vs A-METSK-HD	$< 1E-5$
L-TSK vs FS_{MOGFS}^{e}+TUN^{e}	0.062

IV. CONCLUSIONS

In this paper, the L-TSK (Linguistic TSK) genetic fuzzy system was presented, for learning simple TSK fuzzy rule bases in regression problems. This new approach has two general-purpose preprocessing stages for regression problems: a new instance selection for regression and a novel non uniform multi-granularity fuzzy discretization. The evolutionary learning algorithm incorporates an automatic generation of the TSK fuzzy rule bases from fuzzy partitions that uses Elastic Net in order to obtain consequents with low overfitting.

L-TSK was compared with three state of the art algorithms that learn different types of fuzzy rules: approximative Mamdani, linguistic type-0 TSK and approximative type-1 TSK. The results were analyzed using statistical tests, which show that L-TSK obtains a high accuracy comparable with the approximative TSK approach, but with a lower number of rules. This is of particular interest in problems where both

high accuracy and interpretability are demanded, in order to provide qualitative understanding of the model to the users.

ACKNOWLEDGMENTS

This work was supported by the Spanish Ministry of Economy and Competitiveness under projects TIN2011-22935, TIN2011-29827-C02-02 and TIN2014-56633-C3-1-R, and the Galician Ministry of Education under the projects EM2014/012 and CN2012/151. I. Rodriguez-Fdez is supported by the Spanish Ministry of Education, under the FPU national plan (AP2010-0627).

REFERENCES

[1] T. Hastie, R. Tibshirani, J. Friedman, T. Hastie, J. Friedman, and R. Tibshirani, *The elements of statistical learning*. Springer, 2009.

[2] O. Cordón, F. Herrera, F. Hoffmann, L. Magdalena, O. Cordon, F. Herrera, and F. Hoffmann, *Genetic fuzzy systems*. World Scientific Publishing Company Singapore, 2001.

[3] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Transactions on Systems, Man and Cybernetics*, no. 1, pp. 116–132, 1985.

[4] R. Alcalá, M. J. Gacto, and F. Herrera, "A fast and scalable multiobjective genetic fuzzy system for linguistic fuzzy modeling in high-dimensional regression problems," *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 4, pp. 666–681, 2011.

[5] M. J. Gacto, M. Galende, R. Alcalá, and F. Herrera, "Metsk-hd e: A multiobjective evolutionary algorithm to learn accurate tsk-fuzzy systems in high-dimensional and large-scale regression problems," *Information Sciences*, vol. 276, pp. 63–79, 2014.

[6] A. A. Márquez, F. A. Márquez, A. M. Roldán, and A. Peregrín, "An efficient adaptive fuzzy inference system for complex and high dimensional regression problems in linguistic fuzzy modelling," *Knowledge-Based Systems*, vol. 54, pp. 42–52, 2013.

[7] M. Fazzolari, R. Alcalá, Y. Nojima, H. Ishibuchi, and F. Herrera, "A review of the application of multiobjective evolutionary fuzzy systems: Current status and further directions," *IEEE Transactions on Fuzzy Systems*, vol. 21, no. 1, pp. 45–65, 2013.

[8] S. Garcia, J. Derrac, J. R. Cano, and F. Herrera, "Prototype selection for nearest neighbor classification: Taxonomy and empirical study," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 3, pp. 417–435, 2012.

[9] M. Fazzolari, B. Giglio, R. Alcalá, F. Marcelloni, and F. Herrera, "A study on the application of instance selection techniques in genetic fuzzy rule-based classification systems: Accuracy-complexity trade-off," *Knowledge-Based Systems*, vol. 54, pp. 32–41, 2013.

[10] I. Rodríguez-Fdez, M. Mucientes, and A. Bugarín, "An instance selection algorithm for regression and its application in variance reduction," in *Proceedings of the 2013 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2013.

[11] R. Alcalá, J. Alcalá-Fdez, F. Herrera, and J. Otero, "Genetic learning of accurate and compact fuzzy rule based systems based on the 2-tuples linguistic representation," *International Journal of Approximate Reasoning*, vol. 44, no. 1, pp. 45–64, 2007.

[12] H. Ishibuchi and T. Yamamoto, "Performance evaluation of fuzzy partitions with different fuzzification grades," in *Proceedings of the 2002 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'02).*, vol. 2. IEEE, 2002, pp. 1198–1203.

[13] M. Fazzolari, R. Alcalá, and F. Herrera, "A multi-objective evolutionary method for learning granularities based on fuzzy discretization to improve the accuracy-complexity trade-off of fuzzy rule-based classification systems: D-mofarc algorithm," *Applied Soft Computing*, vol. 24, no. 0, pp. 470 – 481, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1568494614003603

[14] S. Garcia, J. Luengo, J. A. Sáez, V. López, and F. Herrera, "A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 4, pp. 734–750, 2013.

[15] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the Royal Statistical Society*, vol. 67, no. 2, pp. 301–320, 2005.

[16] E. Marchiori, "Class conditional nearest neighbor for large margin instance selection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 2, pp. 364–370, 2010.

[17] J. Dougherty, R. Kohavi, M. Sahami *et al.*, "Supervised and unsupervised discretization of continuous features," in *Machine learning: proceedings of the twelfth international conference*, vol. 12, 1995, pp. 194–202.

[18] D. W. Scott, *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons, 2009, vol. 383.

[19] W. H. Press, *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press, 2007.

[20] F. Herrera and L. Martínez, "A 2-tuple fuzzy linguistic representation model for computing with words," *Fuzzy Systems, IEEE Transactions on*, vol. 8, no. 6, pp. 746–752, 2000.

[21] L.-X. Wang and J. M. Mendel, "Generating fuzzy rules by learning from examples," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 22, no. 6, pp. 1414–1427, 1992.

[22] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of the 19th International Conference on Computational Statistics*. Springer, 2010, pp. 177–186.

[23] Y. Tsuruoka, J. Tsujii, and S. Ananiadou, "Stochastic gradient descent training for l1-regularized log-linear models with cumulative penalty," in *Proceedings of the Joint conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*. Association for Computational Linguistics, 2009, pp. 477–485.

[24] F. Herrera, M. Lozano, and A. M. Sánchez, "A taxonomy for the crossover operator for real-coded genetic algorithms: An experimental study," *International Journal of Intelligent Systems*, vol. 18, no. 3, pp. 309–338, 2003.

[25] J. Alcala-Fdez, L. Sanchez, S. Garcia, M. J. del Jesus, S. Ventura, J. Garrell, J. Otero, C. Romero, J. Bacardit, V. M. Rivas *et al.*, "Keel: a software tool to assess evolutionary algorithms for data mining problems," *Soft Computing*, vol. 13, no. 3, pp. 307–318, 2009.

[26] I. Rodríguez-Fdez, A. Canosa, M. Mucientes, and A. Bugarín, "STAC: a web platform for the comparison of algorithms using statistical tests," 2015. [Online]. Available: http://tec.citius.usc.es/stac