DE GRUYTER
OPEN

JAISCR, 2017, Vol. 7, No. 2, pp.137– 147
10.1515/jaiscr-2017-0010

DESIGN OF FUZZY RULE-BASED CLASSIFIERS THROUGH GRANULATION AND CONSOLIDATION

Andri Riid, Jürgo-Sören Preden

Laboratory for Proactive Technologies, Tallinn University of Technology,
Ehitajate tee 5, 19086, Tallinn, Estonia

Abstract

This paper addresses the issue how to strike a good balance between accuracy and compactness in classification systems - still an important question in machine learning and data mining. The fuzzy rule-based classification approach proposed in current paper exploits the method of rule granulation for error reduction and the method of rule consolidation for complexity reduction. The cooperative nature of those methods - the rules are split in a way that makes efficient rule consolidation feasible and rule consolidation itself is capable of further error reduction - is demonstrated in a number of experiments with nine benchmark classification problems. Further complexity reduction, if necessary, is provided by rule compression.

Keywords: pattern recognition, fuzzy classification, complexity reduction

1 Introduction

Classification is a basic task in performing data analysis or pattern recognition, therefore many problems in very different fields (such as biology, medicine, information retrieval, national security, speech/handwriting recognition, spam filtering etc.) can be represented as classification problems. This explains the need for increasingly accurate, reliable and computationally efficient classifiers [3].

Simply put, a classifier is a function that assigns a class label to an object (observation) on the basis the object description. The latter is given by a vector that contains values of the features or attributes (present paper focuses only on data sets with numerical attributes but the attributes can also be nominal and ordinal) of the object that are considered to be relevant for the classification task. Usually, the classifier is trained to predict class labels using a training algorithm and a training data set. Once the training is completed, the classifier is expected to perform favorably on unseen objects.

There exists a number of classification algorithms including Bayesian classifiers [12], nearest neighbor classifiers [11], rule-based classifiers [9], support vector machines [10], classification trees [6, 26], neural classifiers [8, 23], fuzzy logic-based classifiers [4, 17, 19] and many hybrid and ensemble methods [27, 30].

Fuzzy rule-based classifiers are fuzzy systems specifically configured for performing classification tasks that consist of a number of classification rules and utilize fuzziness only in the reasoning mechanism of the classifier [21]. Often, fuzzy rule-based classifiers are considered more intuitive and interpretable than more common black box classification systems [24]. Although interpretability and accuracy are considered to be contradictive requirements, fuzzy rule-based classifiers are not necessarily less accurate than other classifiers.

Generally speaking, the goal in fuzzy rule-based classification is to obtain the maximum possible classification accuracy with as simple classifier as possible. Classification accuracy that a data-driven fuzzy rule-based classifier is able to achieve, first and foremost depends on the properties of the data set. Instances of classes that are separate from instances of other classes in product space are easy to classify correctly whereas high overlap of classes can make it very difficult to obtain an accurate classifier.

The class distributions that do not separate naturally in product space typically need to be modeled with increased level of granularity. Alternatively, optimal or near-optimal decision border may be provided by suitable rule placement. In this study we propose an approach that explores both these options to yield accurate yet compact classifiers.

The proposed approach includes the following steps: 1) classifier initialization (generation of a minimal rule classifier); 2) rule splitting procedure that gradually increases granularity of the classifier until satisfying accuracy level has been met; 3) rule consolidation that reduces the number of classification rules either maintaining the obtained level of accuracy or even improving on that. These steps of the approach are described in sections 3.1, 3.2 and 3.3, respectively.

The 10-fold cross-validation classification results on nine benchmark datasets provided in section 5 demonstrate the competitiveness of the proposed approach. If desired, complexity of a fuzzy classifier can be further reduced by reducing the number of conditions in the rules (termed as rule compression).

2 Preliminaries

A fuzzy rule-based classifier consists of R rules in the following format

$$\text{IF } x_1 \text{ is } A_{1r} \text{ AND } x_2 \text{ is } A_{2r} \text{ AND ... AND } x_N \text{ is } A_{Nr} \text{ THEN } y \text{ belongs to class } c_r, \quad (1)$$

where A_{ir} denote the linguistic labels of the i-th feature associated with the r-th rule ($i = 1,...,N; r = 1,...,R$) and c_r is a class label assigned to the r-th rule ($c_r \in \{1,...,T\}$). Note that the actual numerical value of c_r is irrelevant, it just functions as a label because class is a nominal variable.

Each A_{ir} has its representation in the numerical domain - a typically normal and convex membership function μ_{ir}. In present study we employ the membership functions (MFs) that are built upon two Gaussian curves defined by the positions of the peaks b_1 and b_2 and standard deviations σ_1 and σ_2, respectively. We assume that $b_1 = b_2 = b$ thus the MF appears as

$$\mu(x) = \begin{cases} e^{-\frac{(x-b)^2}{2\sigma_1^2}}, & x < b \\ e^{-\frac{(x-b)^2}{2\sigma_2^2}}, & x \geq b \end{cases}. \quad (2)$$

From 2 we derive the expressions for σ_1 and σ_2 so that $\mu(a) = \mu(c) = \alpha$, $\alpha < 1$.

$$\alpha = e^{-\frac{(a-b)^2}{2\sigma_1^2}} \Rightarrow \sigma_1 = \sqrt{-\frac{(a-b)^2}{2\ln(\alpha)}}, \quad (3)$$

$$\alpha = e^{-\frac{(c-b)^2}{2\sigma_2^2}} \Rightarrow \sigma_2 = \sqrt{-\frac{(c-b)^2}{2\ln(\alpha)}}. \quad (4)$$

By substituting 3 and 4 into 2 we obtain

$$\mu(x) = \begin{cases} e^{\frac{\ln\alpha(x-b)^2}{(a-b)^2}}, & x < b \\ e^{\frac{\ln\alpha(x-b)^2}{(c-b)^2}}, & x \geq b \end{cases}, \quad (5)$$

which further simplifies into

$$\mu(x) = \begin{cases} \alpha^{(\frac{x-b}{a-b})^2}, & x < b \\ \alpha^{(\frac{x-b}{c-b})^2}, & x \geq b \end{cases}. \quad (6)$$

Note that the parameters of 6 - a, b, c and α - are easier to interpret than the standard deviations of 2 that is evidenced in Figure 1.

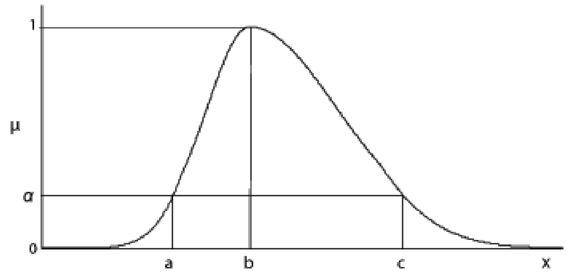

Figure 1. The Gaussian MF employed in current study and the meaning of its parameters.

The reasoning mechanism of a fuzzy rule-based classifier is usually implemented by the single winner approach [1, 16, 18, 20, 24, 31] that selects the class label c_r, associated with the rule that provides the highest rule activation degree (τ_r) for the vector

$\mathbf{x}_k = (x_1(k), x_i(k), ..., x_N(k))$ representing k-th observation.

$$y(k) = c_r, \arg\max_{1 \le r \le R}(\tau_r(k)), \tag{7}$$

where

$$\tau_r(k) = \bigcap_{i=1}^{N} \mu_{ir}(x_i(k)), \tag{8}$$

where $\bigcap_i^N$ represents the minimum operator corresponding to the linguistic operator AND in 1.

3 The proposed method

This Section starts with the description of the rule generation routine that is applied whenever a subset of data is at hand, upon which a classification rule needs to be constructed.

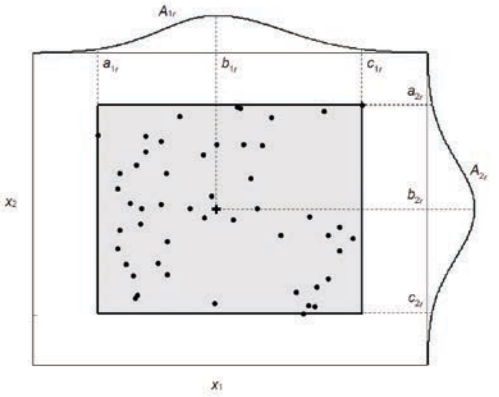

Figure 2. Generation of the r-th rule in product space ($N = 2$). Note that within the rule borders (grey area) $\tau_r > \alpha$ (this is ensured by the minimum operator in refeq:tau).

Given a subset of data S_r that contains K_r observations and its mean $\mathbf{m}_r = (m_{1r}, m_{2r}, ..., m_{Nr})$ that is the geometric centroid of the data points in S_r

$$\mathbf{m}_r = \sum_{k \in S_r} \mathbf{x}_k / K_r, \tag{9}$$

the MFs μ_{ir} of form refeq:mfs are created in all dimensions i. Given a predefined value of α ($\alpha = 0.005$ in all following experiments), the MF parameters a_{ir}, b_{ir}, c_{ir} are obtained as follows. For each i

$$a_{ir} = \min_{k \in S_r}(x_i(k)), c_{ir} = \max_{k \in S_r}(x_i(k)), \\ b_{ir} = m_{ir}. \tag{10}$$

Following this a rule

$$\text{IF } x_1 \text{ is } A_{1r} \text{ AND } x_2 \text{ is } A_{2r} ... \\ \text{AND } x_i \text{ is } A_{ir} ... \text{ AND } x_N \text{ is } A_{Nr} \\ \text{THEN } y \text{ belongs to class } c_r, \tag{11}$$

where A_{ir} represent the MFs μ_{ir} and c_r is the class that is represented by the majority of observations in subset S_r, is constructed (Figure 2).

3.1 Minimal rule classifiers

The simplest classifier possible is the minimal rule classifier (MRC) that specifies only one rule for each class. The training data set is divided into T subsets so that each subset contains only the samples belonging to one of T classes and the rule generation routine refeq:makeMFs-refeq:classst1 is executed until all subsets have been covered. Unless the classes are well separable in the product space, the MRC usually comes with a number of misclassified samples, depending on how “bad” the data is.

It is worth noting that if we replace the designated MFs refeq:mfs with standard Gaussian functions

$$\mu(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-b)^2}{2\sigma^2}}, \tag{12}$$

and use

$$b_{ir} = m_{ir}, \sigma_{ir} = \sqrt{\frac{1}{K_r} \sum_{k \in S_r} (x_i(k) - m_{ir})^2}, \tag{13}$$

in the rule generation process and multiply $\tau_r(k)$ in refeq:classinf by a rule weight $w_r(k)$, which is computed as the prior probability of class c_r samples in training data (the number of samples belonging to class c_r divided by the overall number of samples K) then the MRC what we obtain is, in fact, a Naive Bayesian classifier.

3.2 Rule granulation for error minimization

Each rule of the minimal rule classifier governs a subset of data S_r consisting of K_r samples and usually there is a number of misclassified samples within this subset. The latter figure is denoted by η_r and called local error. The global error (η) is given by

$$\eta = \sum_{r=1}^{R} \eta_r. \tag{14}$$

Classification error reduction is carried out by a sequence of rule splits so that at each iteration a parent rule is selected and split into two offspring rules. The offspring rules replace the parent rule, which means that at each iteration the number of classification rules increases by one. Usually there is a number of choices on which parent rule to pick and how to make the split. The first choice for the parent rule is a rule p with the highest local error

$$p = r, \arg \max_{1 \le r \le R}(\eta_r). \tag{15}$$

If there are several rules with the same local error, we simply choose the one with the highest K_r of those.

The rule splitting cut can be made around each erroneous sample under the parent rule. At given iteration, a single cut is allowed at one of N coordinates, thus the overall number of potential rule splits at the iteration equals $N \times \eta_p$. For example, in the situation depicted in Figure 3, $N = 2$, $\eta_p = 1$ and thus two cuts are possible.

A cut divides the K_p samples of the parent rule into two subsets S_o and S_q that form the basis of two offspring rules, R_o and R_q. Note that the erroneous sample is always sided with the offspring rule that contains less samples. Of available cuts the one that results in the best performing classifier (yielding the least number of η) is selected. It is possible that there are several cuts that result in classifiers with the same number of erroneous samples. In this case we choose the cut that has the minimal value of $\max(\eta_o, \eta_q)$ - generally this leads to faster convergence. If this still leaves us several equally good candidates, we choose the cut that has a smaller value of $\min(K_o, K_q)$.

The splitting continues until η reaches either zero (as in Figure 3, right), some other pre-specified higher value or yet another ending criterion (e.g. pre-set overall accuracy rate, overall number of rules) becomes satisfied.

3.3 Rule consolidation

The procedure for reducing the number of rules of classifiers is outlined in [29] and termed rule base consolidation. During the consolidation, weaker rules (governing few samples) are constantly losing their samples to stronger rules (those governing many samples). Each such sample transfer is valid as long as accuracy of the classifier is not compromised. As a natural result, many of the weaker rules become obsolete.

The rules are ranked by their strength (the number of samples they govern) in ascending order $p \in \{1, ..., R\}$. The process starts from the lowest ranked rule ($p = 1$):

1. Pick a rule R_r with the rank p.
2. Pick k-th sample ($k = 1, ..., K_r$) from the subset S_r governed by rule R_r.
3. Transfer this sample from S_r to the subset S_q corresponding to R_q, the next rule in the ranking that matches the class of the sample ($c_q = y_k$).
4. Update the MFs of both R_r and R_q on the basis of modified subsets S_r and S_q, respectively.

It is then verified if the accuracy loss due to consolidation has occurred (the global error of the consolidated classifier has increased). If there is no accuracy loss, the tranfer is accepted. Otherwise, the transfer is rejected. Based on this decision we proceed as follows:

- If the transfer is accepted and $k < K_r$, increment k (select the next sample from S_r). If k, however, already equals K_r, delete rule R_r along with associated MFs, update the ranking, increment p and go back to step 1.
- If the transfer is rejected, first discard the changes to the MFs of R_r and R_q, pick the next matching rule from the ranking and go back to step 3. If we already have reached the last matching rule in the ranking, select the next sample from subset S_r (increment k) and go to step 2. If k already equals K_r as well, increment p and return to step 1.

The process comes to an end when we have reached the last rule in the ranking ($p = R$). It can be, however, started over from the beginning and carried on until the consolidation stabilizes (i.e. there are no more accepted transfers).

4 Results

All nine data sets that have been chosen for classification experiments and verification of the pro-

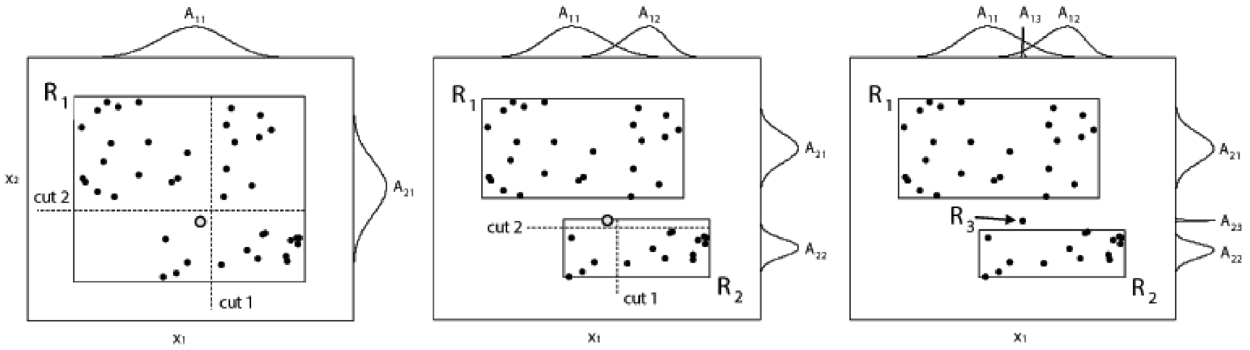

Figure 3. Rule granulation. With other criteria being equal, cut 2 in the left graph is preferred over cut 1 because the offspring rule R_2 governs less samples than any of involved rules in the alternative scenario. Cut 2 in the middle graph is preferred over cut 1 because it promptly reduces the classification error to zero.

posed method, are well-known, feature frequently in classification and pattern recognition studies and are available through the UCI Machine Learning Repository [5]. These are the Iris [14], Wine [2], Thyroid [25], Glass [13], BUPA Liver Disorders [34], Cleveland Heart Disease [15], Pima Indian Diabetes [32] and two variants of Wisconsin Breast Cancer data sets (referred to as WDBC [33] and Breast Cancer [35], respectively).

The MRCs for those classification problems are given in Table 1.

Table 1. MRC results on benchmark problems.

Data set name	N	T	K	ε (%)	η
Iris	4	3	150	94.67	8
Wine	13	3	178	95.51	8
Thyroid	21	3	3772	99.89	4
WDBC	30	2	569	95.61	25
Breast Cancer	9	2	683	58.27	285
Glass	9	6	214	58.88	88
Bupa	6	2	345	59.13	141
Cleveland	13	5	297	24.58	224
Pima	8	2	768	55.99	338

The table shows the number of features (N), the number of classes (T) and the number of samples (K) for each data set, as well as the number of misclassified samples (η) and overall accuracy (ε) of corresponding MRCs. The latter is computed as

$$\varepsilon = 1 - \eta/K, \tag{16}$$

and we can see that for several classification problems listed in Table 1, accuracy of MRCs is not particularly high.

Table 2. Comparison of simple classifiers on benchmark problems.

Data set name	T	MRC	NBC	CART
Iris	3	96.00	96.00	96.00
Wine	3	95.51	**98.88**	88.76
Thyroid	3	**99.89**	92.57	97.91
WDBC	2	**95.61**	94.02	92.27
Breast Cancer	2	94.28	**96.34**	92.68
Glass	6	**72.43**	54.67	71.03
Bupa	2	**65.80**	55.94	63.19
Cleveland	5	59.93	**62.29**	59.60
Pima	2	67.19	**76.17**	73.57

Rule consolidation algorithm alone is often able to improve the classification accuracy because it can transfer the misclassified samples to properly labelled rules. For example, when rule consolidation is applied to the MRCs from Table 1, it reduces the initial classification error considerably for some, previously ill-classified data sets. This way, classification accuracy for the Iris data set increases from 94.67% to 96.00% (6 erroneous samples); from 58.27% to 94.29% (39 erroneous samples) for Breast Cancer data set; from 59.13% to 65.80% (118 erroneous samples) for BUPA data set; from 24.58% to 59.93% (119 erroneous samples) for Cleveland Heart Disease data set; from 55.99% to 67.19% (252 erroneous samples) for Pima Indian Diabetes data set and from 58.88% to 72.43% (59 erroneous samples) for the Glass data set.

Comparison between the consolidated MRCs, Naive Bayesian classifiers and decision trees (CART, [6]) with T leaf nodes is given in Table 2. The best result for each data set is highlighted.

Figure 4 plots the error curves in blue throughout the rule granulation steps for all benchmark data sets. Depending on the initial error and data set properties it takes a varying number of splits to obtain the minimum error classifier (MEC) at the last split. Broadly speaking, the data sets in the first row are the easy ones, the data sets in the second row are more difficult and the ones in the last row present a challenge. Typically, we can see a rapid improvement of classification accuracy in the beginning of the training process (which does not last long). This is followed by a more hectic or stagnation phase where there is almost no improvement (the latter phenomenon is particularly clearly evidenced in the graph related to the Breast Cancer data set) just further fragmentation of rules. In the last phase of the training, however, the improvement is slow but steady (this is because at this point the rules that contain errors are small).

Table 3. the number of rules of benchmark data set MECs before (R_s) and after consolidation (R_c) and the number of leaf nodes of decision trees describing the same problem (n_l).

Data set name	R_s	R_c	n_l (CART)
Iris	10	7	9
Wine	9	4	12
Thyroid	10	6	13
WDBC	24	7	22
Breast Cancer	40	12	32
Glass	51	22	50
Bupa	104	35	80
Cleveland	124	42	101
Pima	202	54	128

Table 3 contains the number of rules of MECs after rule granulation (R_s) and after consolidation (R_c). Comparable complexity measure, i.e. the number of leaf nodes (n_l) of a CART applied for the same classification problem is added for reference. From this comparison we can see that while for some data sets n_l can be smaller than R_s, it is always larger than R_c.

The green curve in Figure 4 indicates the corresponding accuracy rates of the classifiers to which instantaneous consolidation is applied after each split. We can see that when rule consolidation is applied to an erroneous classifier it typically gives a significant boost in accuracy, especially for “difficult” data sets. The red curve tied to the second y-axis in these graphs depicts the number of rules after each consolidation operation.

In practice, however, we do not need so much a flawless and possibly overtrained classifier with many parameters, rather than a compact one that would capture the essence of the classification problem. For this we should be able to guess the breaking point or the “soft spot” in learning located somewhere in the second phase when the algorithm is turning its attention to the erroneous samples, which, for all we know could be just measurement errors or outliers in the data.

5 Performance on unseen data

In previous Sections we have shown that using the combination of rule granulation and consolidation, it is possible to obtain the classifiers of arbitrary accuracy (on training data, that is). In practice, however, the ability of a classifier to learn the training data is less important than its generalization ability, i.e its ability to predict the class labels for new, unseen samples of data. This ability is usually estimated using 10-fold cross validation by which the original data set is randomly divided into 10 disjoint sets (folds) of equal size where each fold has roughly the same class distribution. The classifier is trained 10 times, each time with a different set held out as a test set and the other 9 subsets put together to form a training set. This way each data point gets to be in a test set exactly once and in a training set 9 times. In the end, the mean values of training and testing accuracies across all 10 trials are computed that serve as the performance measures.

We have performed cross validation on four types of classifiers, namely: Naive Bayesian classifiers, consolidated MRCs, classifiers obtained with the proposed granulation-consolidation method and CARTs.

Note that in order to avoid overfitting in the proposed method, the rules are split until the highest local error has come down to a pre-specified value η_{min} that is roughly correlated to the initial training error. For Iris, Wine and Thyroid data sets, thus $\eta_{min} = 1$, for WDBC, Breast Cancer and Glass data sets, $\eta_{min} = 3$, for BUPA and Cleveland data sets $\eta_{min} = 5$ and for the Pima data set $\eta_{min} = 10$. The

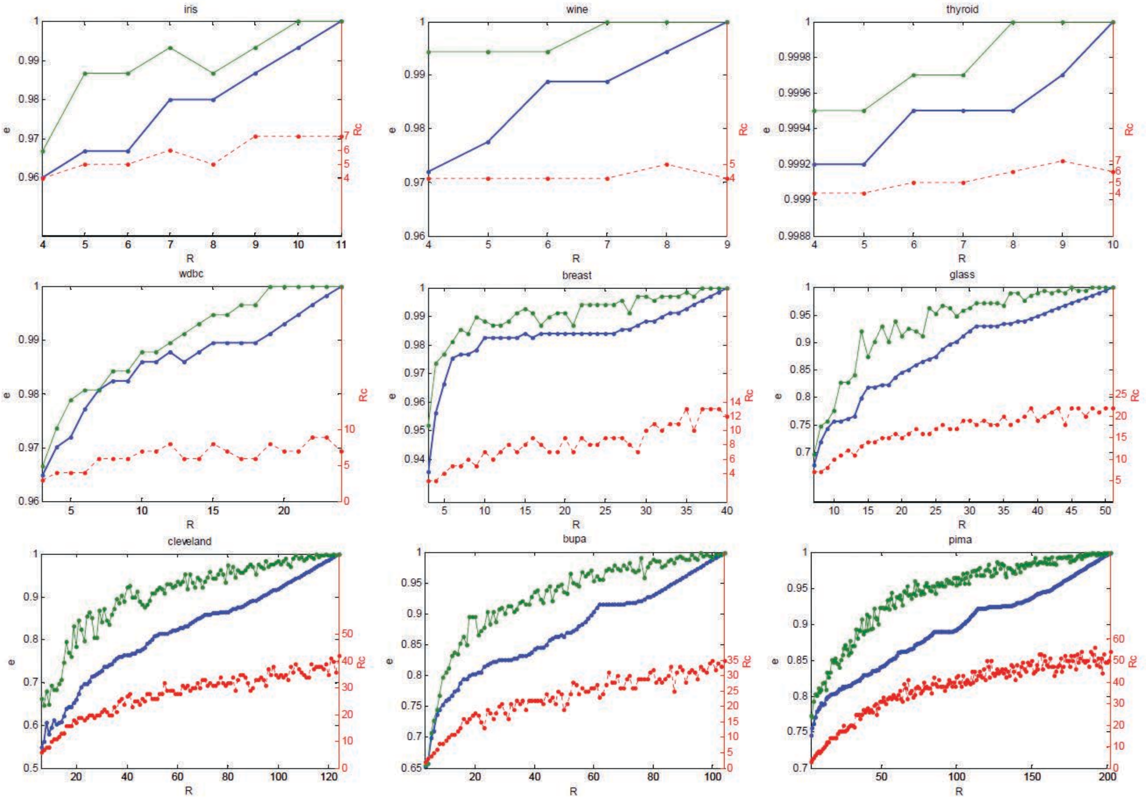

Figure 4. The learning curves of the proposed approach on nine benchmark data sets with and without rule consolidation (green and blue lines, respectively). The number of consolidated rules at each training step is depicted by a red curve.

Table 4. Stratified 10-fold cross validation results on benchmark data sets.

Data set	NBC			MRC		proposed method				CART		
	R	ε_{tr}	ε_{tst}	ε_{tr}	ε_{tst}	R_s	R_c	ε_{tr}	ε_{tst}	n_l	ε_{tr}	ε_{tst}
Iris	3	96.07	95.33	95.85	94.67	5.5	4.4	98.30	**96.67**	4.7	98.15	95.33
Wine	3	98.44	**96.66**	95.88	91.63	5.2	4.0	99.62	95.00	8.1	99.19	90.52
Thyroid	3	92.59	92.67	99.81	99.42	5.3	4.0	99.94	99.44	11.2	99.98	**99.66**
WDBC	2	93.91	93.33	95.72	93.85	8.4	5.5	98.63	**94.03**	8.3	97.85	93.68
BC	2	96.24	**96.20**	92.24	92.39	8.5	5.7	98.75	95.32	9.6	97.69	95.31
Glass	6	54.41	50.87	68.70	53.75	20.1	13.9	93.82	**65.46**	19.2	87.07	62.73
BUPA	2	57.46	54.23	64.38	59.50	22.9	16.1	90.60	**69.56**	19.5	84.35	68.96
Cleveland	5	63.63	**55.13**	61.58	53.98	29.9	18.8	86.04	54.86	26.4	78.04	53.22
Pima	2	76.45	**75.26**	69.46	67.97	27.3	19.8	89.19	**75.26**	22.5	85.32	73.83

same values of η_{min} serve as the node splitting stop criteria for individual data sets in a CART - a node is no longer split if it contains η_{min} (or less) erroneous samples. Also note that the data folds are fixed for all compared algorithms to obtain comparable classification results.

The results of the validation are given in Table 4 where ε_{tr} and ε_{tst} denote the training and testing accuracies, respectively, and the best testing accuracy for each data set is highlighted.

The cross-validation results partially confirm the frequent claim that "Naive Bayes can often outperform more sophisticated classification methods" [22] as Naive Bayes appears to be winner in 3 out of 9 cases and ties once with the proposed method. When it loses, however, it can lose by a large margin as the results on Glass, BUPA and Thyroid data sets indicate.

On the other hand, the proposed method is the overall winner in still more cases, outperforms CART in terms of testing accuracy in 8 out of 9 cases and the number of fuzzy rules after consolidation (R_c) is always smaller than the corresponding number of leaf nodes (n_l). By definition, however, the attribute tests are applied only to a handful of available attributes in the internal nodes that are in the path from the root node to a leaf node in a CART, which would correspond to a fuzzy rule having a limited number of conditions (so called incomplete rules).

To reduce the number of conditions in fuzzy classifiers we apply the naive rule compression method [28]. The algorithm is based on simple trial and error and is described as follows:

1. Pick the rule R$_r$ ($r = 1,...,R$).
2. Rank the features $i = 1,...,N$ by MF spread ($c_{ir} - a_{ir}$), normalized, of course, in descending order (this way the features in which the subset of samples governed by r-th rule is less compact, less defined, are removed first).
3. Discard the conditions applied to the features one by one, in the order of ranking, reversing those removals that would result in loss of accuracy.

The average numbers of conditions before and after compression - n_{cond} and n^*_{cond}, respectively - of fuzzy classifiers are given in Table 5 along with cross-validated classification accuracy before and after compression (values of ε_{tst} are lifted from Table 4). The adjusted number of attribute tests, i.e. the total number of internal nodes met on n_l individual paths from the root node to all leaf nodes in a CART (tests concerning the same attribute in a given path are taken into account only once) is added for reference. One can see that the number of conditions is below the number of attribute tests but this is partially so because n_l tends to be larger than R_c. The average number of conditions per rule in a classifier, however, is only 10% higher than the average number of attribute tests per leaf node, which is good enough.

Fuzzy rules with less conditions appear to be more general, and thus one would expect them to have more generalizational power, however, as our results indicate, this is true only for half the cases.

To put this piece of research into context, one can note that the obtained testing accuracies are in the same ballpark with figures available from literature. For example, [7] that lists the results of various algorithms (Support Vector Machines, associative rules, Naive Bayesian classifiers and decision trees) reports the testing accuracies of 94.5±2.2% for Iris, 96.45±1.35% for Breast Cancer, 65.95±5.05% for BUPA and 75.55±2.65% for Pima data sets. In a more recent study [4] with the results of four genetic fuzzy classifiers, four associative classifiers and one decision tree (C4.5), the corresponding figures are 94.65±1.35% for Iris, 93.08±1.62% for Wine, 74.08±1.58% for Pima, 52.86±4.04% for Cleveland, 92.96±2.29% for WDBC and 64.65±6.65% for Glass data sets.

Conclusions

The primary goals in fuzzy rule-based classification are accuracy maximization and complexity minimization, which are contradicting requirements. The approach presented in current paper - a combination of rule granulation and rule consolidation methods and further rule compression - offers the possibility to find a good compromise between those requirements.

A properly designed classification algorithm must have good learning abilities to be able to dis-

Table 5. Cross validation results on compressed classifiers.

Data set name	proposed method				CART	
	ε_{tst}	ε^*_{tst}	n_{cond}	n^*_{cond}	ε_{tst}	n_{cond}
Iris	96.67	**97.33**	17.6	8.8	95.33	8.4
Wine	**95.00**	93.36	52.0	13.7	90.52	25.7
Thyroid	99.44	99.55	81.9	14.6	**99.66**	50.3
WDBC	**94.03**	93.50	165.0	25.0	93.68	25.5
Breast Cancer	95.32	**95.76**	51.3	22.3	95.31	29.0
Glass	65.46	**69.55**	125.1	51.1	62.73	94.7
BUPA	**69.56**	66.65	96.6	67.2	68.96	75.3
Cleveland	54.86	**56.88**	244.4	100.7	53.22	127.2
Pima	**75.26**	74.09	158.4	101.6	73.83	91.6

cover patterns in data, which is valid for the proposed method but the true criterion of a good classifier is its predictive performance, estimated by cross-validation. However, there is no single classification algorithm that is best for all types of data. A method can outperform others on an almost consistent basis and yet show weaker performance on certain data sets. This is apparent in present study as well as of the chosen benchmarks, the proposed method performs best on Iris, WDBC, Glass, BUPA and Cleveland data sets and ties with Naive Bayesian classifier on Pima data set, whereas Naive Bayesian classifier performs best on Wine and Breast Cancer data sets. CART, on the other hand, outperforms other algorithms on Thyroid data set.

Acknowledgements

The work presented in this paper was partially supported by the Estonian Ministry of Defence and the European Defense Agency project IN4STARS 2.0.

References

[1] J. Abonyi, J. A. Roubos, and F. Szeifert, Data-driven generation of compact, accurate, and linguistically sound fuzzy classifiers based on a decision-tree initialization, International Journal of Approximate Reasoning, 23:1–21, 2003

[2] S. Aeberhard, D. Coomans, and O. de Vel, Comparative analysis of statistical pattern recognition methods in high dimensional settings, Pattern Recognition, 27(8):1065–1077, 1994

[3] C. C. Aggarwal, Data Classification: Algorithms and Applications, Chapman & Hall/CRC, 1st edition, 2014

[4] J. Alcala-Fdez, R. Alcala, and F. Herrera, A fuzzy association rule-based classification model for high-dimensional problems with genetic rule selection and lateral tuning, IEEE Transactions on Fuzzy Systems, 19(5):857–872, 2011

[5] K. Bache and M. Lichman, UCI machine learning repository, http://archive.ics.uci.edu/ml, 2013

[6] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, Classification and regression trees, Wadsworth & Brooks/Cole Advanced Books & Software, Monterey, 1984

[7] B.-C. Chien, J.-Y. Lin, and W.-P. Yang, A classification tree based on discriminant functions, Journal of Information Science and Engineering, 222(3):573–594, 2006

[8] S.-B. Cho, Neural-network classifiers for recognizing totally unconstrained handwritten numerals, IEEE Transactions on Neural Networks, 8(1):43–53, 1997

[9] W. W. Cohen, Fast effective rule induction, In Proceedings of the Twelfth International Conference on Machine Learning, pages 115–123, Morgan Kaufmann, 1995

[10] C. Cortes and V. Vapnik, Support-vector networks, Machine Learning, 20(3):273–297, September 1995

[11] T. Cover and P. Hart, Nearest neighbor pattern classification, IEEE Transactions on Information Theory, 13(1):21–27, September 2006

[12] R. Duda and P. Hart, Pattern Classification and Scene Analysis, Wiley, New York, 1973

[13] I. W. Evett and E. J. Spiehler, Rule induction in forensic science, Technical report, Central Research Establishment, Home Office Forensic Science Service, 1987

[14] R. A. Fisher, The use of multiple measurements in taxonomic problems, Annals of Eugenics, 7(2):179–188, 1936

[15] J. H. Gennari, P. Langley, and D. Fisher, Models of incremental concept formation, Artificial Intelligence, 40(1–3):11–61, 1989

[16] S. Guillaume and B. Charnomordic, Learning interpretable fuzzy inference systems with FisPro, Information Sciences, 180(20):4409–4427, 2011

[17] J. Hühn and E. Hüllermeier, FURIA: an algorithm for unordered fuzzy rule induction, Data Mining and Knowledge Discovery, 19(3):293–319, 2009

[18] H. Ishibuchi, T. Nakashima, and T. Murata, Three-objective genetic-based machine learning for linguistic rule extraction, Information Sciences, 136(1–4):109–133, 2001

[19] H. Ishibuchi, K. Nozaki, N. Yamamoto, and H. Tanaka, Selecting fuzzy if-then rules for classification problems using genetic algorithms, IEEE Transactions on Fuzzy Systems, 3(3):260–270, 1995

[20] H. Ishibuchi and T. Yamamoto, Rule weight specification in fuzzy rule-based classification systems, IEEE Transactions on Fuzzy Systems, 13(4):428–435, 2005

[21] L. Kuncheva, Fuzzy Classifier Design, Springer-Verlag, Heidelberg, 2000

[22] K. Larsen, Generalized naive bayes classifiers, SIGKDD Explorations, 7(1):76–81, 2005

[23] R. P. Lippmann, Neural network classifiers for speech recognition, The Lincoln Laboratory Journal, 1:107–124, 1988

[24] C. Mencar, C. Castiello, R. Cannone, and A. M. Fanelli, Interpretability assessment of fuzzy knowledge bases: A cointension based approach, International Journal of Approximate Reasoning, 52(4):501–518, 2011

[25] J. R. Quinlan, Induction of decision trees, Machine Learning, 1(1):81–106, 1986

[26] J. R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993

[27] Y. Ren, L. Zhang, and P. N. Suganthan, Ensemble classification and regression - recent developments, applications and future directions, IEEE Computational Intelligence Magazine, 11(1):41–53, 2016

[28] A. Riid and J.-S. Preden, Interpretability improvement of fuzzy rule-based classifiers via rule compression, In Proceedings of the 2015 Conference of the International Fuzzy Systems Association and the European Society for Fuzzy Logic and Technology, pages 162–169, Gijon, Spain, 2015

[29] A. Riid and M. Sarv, Determination of regional variants in the versification of estonian folksongs using an interpretable fuzzy rule-based classifier, In Proceedings of the 8th Conference of the European Society for Fuzzy Logic and Technology (EUSFLAT 2013), pages 61–66, Milan, Italy, 2013

[30] L. Rokach, Pattern Classification Using Ensemble Methods, volume 75 of Series in Machine Perception and Artifical Intelligence, World Scientific Publishing Company, Singapore, 2010

[31] H. Roubos, M. Setnes, and J. Abonyi, Learning fuzzy classification rules from data, Information Sciences, 150(1–2):77–93, 2003

[32] J. W. Smith, J. E. Everhart, W. C. Dickson, W. C. Knowler, and R. S. Johannes, Using the ADAP learning algorithm to forecast the onset of diabetes mellitus, In Proceedings of the Symposium on Computer Applications and Medical Care, pages 261–265, Los Alamitos, CA, 1988

[33] W. N. Street, W. H. Wolberg, and O. L. Mangasarian, Nuclear feature extraction for breast tumor diagnosis, In Proceedings of the IS&T 1993 International Symposium on Electronic Imaging: Science and Technology, volume 1905, pages 861–870, San Jose, CA, 1993

[34] P. D. Turney, Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm, Journal of Artificial Intelligence Research, 2:369–409, 1995

[35] W. H Wolberg and O. L. Mangasarian, Multisurface method of pattern separation for medical diagnosis applied to breast cytology, Proceedings of the National Academy of Sciences, 87:9193–9196, 1990

Andri Riid received his M.Sc. and Ph.D. degrees in System Engineering from Tallinn University of Technology in 1997 and 2002, respectively. He currently works as a Senior Research Scientist in the Laboratory for Proactive Technologies of the same university. His research interests include properties of fuzzy systems and development of algorithms for fuzzy control, modeling and classification.

Jürgo-Sören Preden received his Ph.D. degree in System Engineering from Tallinn University of Technology in 2010. He is currently a Senior Research Scientist and Head of the Research Laboratory for Proactive Technologies at Tallinn University of Technology and CEO of Thinnect. His research interests include distributed computing systems, specifically the situation awareness of such systems. He is a member of IEEE.