Pattern Recognition 46 (2013) 2770–2782

Contents lists available at SciVerse ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

FRPS: A Fuzzy Rough Prototype Selection method

Nele Verbiest [a,]*, Chris Cornelis [a,b], Francisco Herrera [b]

[a] Ghent University, Department of Applied Mathematics, Computer Science and Statistics, Krijgslaan 281 S9, 9000 Gent, Belgium
[b] University of Granada, Department of Computer Science and Artificial Intelligence, E-18071 Granada, Spain

ARTICLE INFO

Article history:
Received 29 February 2012
Received in revised form
26 February 2013
Accepted 5 March 2013
Available online 15 March 2013

Keywords:
Classification
Fuzzy rough sets
Instance selection
k NN
Prototype Selection

ABSTRACT

The *k* Nearest Neighbour (*k* NN) method is a widely used classification method that has proven to be very effective. The accuracy of *k* NN can be improved by means of Prototype Selection (PS), that is, we provide *k* NN with a reduced but reinforced dataset to pick its neighbours from. We use fuzzy rough set theory to express the quality of the instances, and use a wrapper approach to determine which instances to prune. We call this method Fuzzy Rough Prototype Selection (FRPS) and evaluate its effectiveness on a variety of datasets. A comparison of FRPS with state-of-the-art PS methods confirms that our method performs very well with respect to accuracy.

1. Introduction

Classification methods aim to predict the class $d(t)$ of a new target instance t, based on the knowledge in the given decision system (the training data X). That is, the attribute values $a_1(t),\ldots,a_m(t)$ of t are given and $d(t)$ needs to be determined, making use of the instances X in the decision system and their attribute and class values.

Many classification methods are available. In this work, we focus on the use of k-Nearest Neighbours (k NN, [1]). It determines the k instances in X closest to t and then assigns t to the class that is best represented among these k neighbours. In case of ties, a class is assigned at random from the candidate classes.

k NN is a simple classification method that does not impose assumptions on the data. Due to its local nature it has low bias; more specifically, the error rate of 1NN asymptotically never exceeds twice the optimal Bayes error rate [2]. On the other hand, the local nature also results in a high variance, that is, k NN is highly susceptible to noisy data [3]. Furthermore, k NN needs high storage requirements and has low efficiency caused by multiple computations of similarities between the test and training samples.

A technique that deals with these weaknesses of k NN is Prototype Selection (PS, [4]). It first selects a subset of instances $S\subseteq X$ and then classifies a new instance t using the k NN rule acting over S instead of over X. PS should not be confused with instance selection [5]. Instance selection methods are designed to serve as a general data reduction technique for all kinds of machine learning methods, whereas PS methods are instance selection methods specifically designed to improve k NN classification.

Rough set theory [6], initiated by Pawlak in the early 80s, is a mathematical approach that deals with imperfect knowledge. It has been used widely for feature selection [7–16]. Extending rough sets to fuzzy rough sets [17] and using them for feature selection has been explored extensively [18–28], but using fuzzy rough sets for instance selection is still in its infancy.

A preliminary attempt to use fuzzy rough sets for instance selection can be found in [29], presenting the Fuzzy Rough Instance Selection (FRIS) technique. It uses fuzzy rough set theory to express for each instance its membership to the fuzzy positive region, that is, the extent to which instances indiscernible from it belong to the same class. Only instances belonging to the positive region more than a certain threshold are retained. As we will discuss in Section 2.2, FRIS has some shortcomings.

The aim of this paper is to present a new PS method based on fuzzy rough set theory that we call Fuzzy Rough Prototype Selection (FRPS). First, the instances are ordered according to a measure based on fuzzy rough set theory that evaluates the lack of predictive ability of the instances, and the instances for which the value exceeds a certain threshold are removed from the training set. To determine this threshold, we consider the values of all instances and use each of them as threshold. The final threshold is the threshold for which applying 1NN to the corresponding reduced training set results in the highest training accuracy.

In order to make our method more robust, we replace the strict max operator in the fuzzy rough measure by the Ordered

* Corresponding author. Tel.: +32 9 264 47 72; fax: +32 9 264 49 95.
E-mail addresses: Nele.Verbiest@Ugent.be (N. Verbiest), chriscornelis@ugr.es (C. Cornelis), herrera@decsai.ugr.es, herrera@ugr.es (F. Herrera).

0031-3203/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.patcog.2013.03.004

Weighted Average (OWA) operator. These aggregation operators, introduced by Yager in [30], associate weights to the ordered positions of the values and can hence be used to generalize the max operator to a more robust operator.

The remainder of this work is structured as follows: in Section 2, we summarize the related work on PS methods and fuzzy rough approaches to data reduction. In Section 3, we introduce four versions of our new algorithm, FRPS. In Section 4 we first select the best performing method among these four versions and then demonstrate its good performance by applying it on 58 real datasets from the KEEL dataset repository, as in [31], and compare it to 21 state-of-the-art PS methods and FRIS. Finally, we conclude in Section 5.

2. Related work

In this section we briefly present research related to the FRPS method. In Section 2.1, we review the literature on Prototype Selection methods, while in Section 2.2, we discuss data reduction techniques based on (fuzzy) rough set theory.

2.1. Prototype Selection

In [31], an extensive taxonomy on PS methods can be found. In this section we summarize the conclusions of that paper, recall the FRIS algorithm and position our new approach FRPS in the taxonomy.

2.1.1. Type of selection

Below, we list three types of PS methods that can be distinguished based on the sort of instances they select, together with some important representatives.

1. A first class of techniques are *editing methods*. The main goal of these techniques is not to reduce the size of the decision system, but to improve the classification quality of the k NN rule by removing noisy instances. A simple example of such a technique is Edited Nearest Neighbours (ENN, [32]). It considers every instance in the training set and removes it whenever the class predicted by using the k NN rule over the other instances in the training set is different from its true class. Methods derived from ENN include the Modified Edited Nearest Neighbour (MENN, [33]) method and the All k Nearest Neighbour (AllKNN, [34]) method. One of the most effective editing techniques is the Relative Neighbourhood Graph (RNG, [35]) method. The general idea is that after construction of a proximity graph, instances misclassified by their neighbours in this graph are removed. Another editing technique is the Model Class Selection (MoCS, [36]) method that uses a feedback system to incorporate knowledge about the dataset in a tree-based classifier.
2. *Condensation techniques* try to remove superfluous instances. In general, these methods are good at reducing the dimensionality of the decision system. A well-known condensation technique designed specifically for 1NN is Condensed Nearest Neighbours (CNN, [37]). This technique starts off with an empty set $S=\phi$. Then it runs through all instances in the training set and adds an instance to S if it is wrongly classified when applying the 1NN rule over the current set of instances S. As a result, all instances in the decision system will be classified correctly when applying 1NN over S. A more advanced technique is the Reduced Nearest Neighbour (RNN, [38]) technique. This technique first applies CNN to the entire training set X, resulting in a subset $S \subseteq X$. Next, all instances $x \in S$ are considered iteratively. The instance x is temporarily removed from S and it is verified whether all instances in X are classified correctly when applying the 1NN rule over the subset S. If at least one instance is classified incorrectly, x is re-added to S, otherwise, x is removed from S. This is repeated until all instances $x \in S$ have been considered. Other methods derived from CNN are the Fast Condensed Nearest Neighbour (FCNN, [39]) and Modified Condensed Nearest Neighbour (MCNN, [40]) method. Patterns by Ordered Projections (POP, [41]) finds patterns in the training dataset without calculating distances and eliminates instances not satisfying these patterns. Modified Selective Subset (MSS, [42]) retains a consistent subset of instances such that for each instance in the original training set, there is an instance in this subset closer than any other instance. Reconsistent [43] aims to replace neighbouring instances by a single instance.
3. Finally, *hybrid techniques* aim to simultaneously remove noisy and superfluous instances. They are designed to reduce the dimensionality of the decision system and meanwhile improve the classification using the k NN rule. Many of these techniques are based on evolutionary algorithms. For instance, the Generational Genetic Algorithm (GGA, [44,45]), Random Mutation Hill Climbing (RMHC, [46]), Steady-State Memetic Algorithm (SSMA, [47]) and CHC Evolutionary Algorithm (CHC, [48]) are genetic algorithms where the chromosomes correspond to the instances currently selected, and the fitness function depends both on the current reduction rate and the accuracy of the k NN rule over the current chromosome. The Hit Miss Network Edition Iterative (HMNEI, [49]) is a non-evolutionary hybrid PS algorithm. It represents the decision system as a hit and miss network, for which the structural properties correspond to properties of the instances related to the decision of the k NN rule, such as being a noisy or superfluous instance. The Decremental Reduction Optimization Procedure (DROP, [50]) removes instances if this does not cause a decrease of the training accuracy of the current (reduced) training set. The Class Conditional Instance Selection (CCIS, [51]) method introduces the class conditional nearest neighbour to remove instances. C-Pruner [52] computes the order in which instances should be removed and then removes them if this does not result in a drop of training accuracy. The Instance Based 3 (IB3, [53]) method uses a wait and see evidence gathering method to determine which of the saved instances are expected to perform well during classification. Iterative Case Filtering (ICF, [54]), starts off with the ENN algorithm and then employs neighbours and associates to smooth the decision boundaries.

2.1.2. Evaluation of search

Besides labelling PS methods based on the kind of instances they remove, one can also distinguish between filter and wrapper methods.

In the context of PS methods, *filter techniques* use the k NN rule to decide for partial data if they should be removed or added to the selected instances. CNN is such a filter method: an instance is selected if the 1NN rule over the current subset of instances classifies it wrong. ENN is also a filter method: an instance is removed when k NN applied over the universe of instances classifies it incorrectly.

Wrapper methods on the other hand use the k NN rule for the complete training set: many subsets of instances are generated, and each subset is evaluated using a leave-one-out validation scheme. That is, given a subset of instances S, each instance x in the training set X is classified as follows: In case the instance x is in S, the k NN rule is applied over S without the instance x, that is, the neighbours of x are looked up in S but have to be different from x itself. In case x is not in S, the k NN rule is applied over the

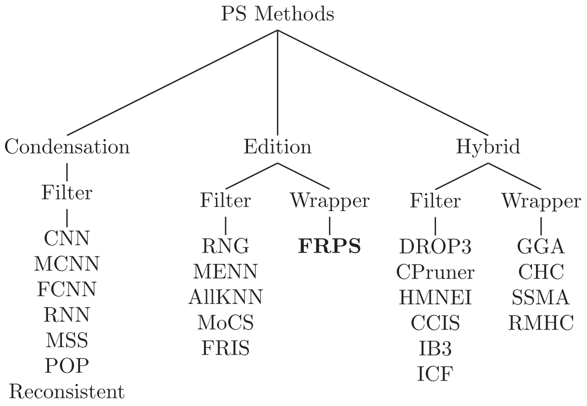

Fig. 1. Taxonomy of PS methods, including FRPS.

entire set S. The accuracy of this classification is then used to evaluate the subset of instances S. Examples of such wrapper methods are GGA and CHC.

As for each considered subset the k NN rule is applied to classify each instance in the training dataset, wrapper methods generally are computationally more expensive than filter methods. On the other hand, wrapper methods make full use of the k NN classifier model and are as a result typically more accurate.

2.1.3. Position of FRPS in the PS taxonomy

In Fig. 1, an overview of the PS taxonomy with the representative examples can be found, including FRPS.

The main goal of the FRPS algorithm is to improve the accuracy of the k NN algorithm by removing instances without good predictive ability. Hence, it can be considered as an editing method.

We want to achieve a good accuracy by using a wrapper method. On the other hand, we want to keep the running time of the algorithm low. That is, we want to achieve an accuracy that is better than that of the existing PS methods, such that the running time is better than that of the hybrid PS methods (that have the advantage of having a better reduction rate) and better than the running time of the most accurate editing PS methods.

2.2. Fuzzy rough set based approaches to data reduction

In this section we discuss data reduction techniques based on fuzzy rough set theory. Most of the work done in this field is on feature selection.

Rough set theory [6] is an excellent tool for feature selection, as it can express how features can discern between classes. There are many ways to use rough set theory for feature selection, mostly evolving around the notion of decision reducts, i.e., minimal subsets of features that preserve the decision making power of the original set of features.

One drawback of classical rough sets is that they cannot appropriately handle continuous features, because rough set theory assumes a crisp indiscernibility relation. One option is to use discretization, but this comes with a loss of information. Therefore, in [17], fuzzy rough sets where introduced, combining fuzzy set theory [55] and rough set theory to obtain a hybrid model. Fuzzy rough sets have been extensively used for feature selection. Many approaches focus on redefining the instances' indiscernibility [21,24,56], while others focus on the partial membership of instances to the classes of a fuzzy partition [19,20,22,23,25,26,57,58] or by extending the definition of decision reduct to fuzzy decision reduct [18].

To the best of our knowledge, only one fuzzy rough approach to instance selection, called Fuzzy Rough Instance Selection (FRIS, [29]), has been proposed. The three variants of the FRIS algorithms described in [29] use the fuzzy rough positive region to decide if an instance should be retained or removed.

- The basic algorithm, *FRIS*-I, deletes all instances for which the positive region membership is lower than a certain threshold (typically 0.95 or 1). One weakness of this algorithm is that it strongly depends on the fuzzy indiscernibility relation and threshold used. In Section 4.2, we will experimentally show that FRIS-I does not perform very well.
- Another variant is *FRIS*-II, which iteratively uses the positive region information to select the object with lowest membership to the positive region for removal and then recalculates each object's membership to the positive region with this object removed.
- The last variant, *FRIS*-III, performs a backward elimination of instances: at every step, it removes the instance whose removal expands the positive region the most, and repeats this until all the instances belong maximally to the positive region.

Unfortunately, both FRIS-II and FRIS-III have time complexity $\mathcal{O}(mn^4)$ if m is the number of attributes and n the number of instances. Hence, these approaches require too much running time to use them in practice. Moreover, the experimental study in [29] shows that there are no significant differences in accuracy between FRIS-I, FRIS-II and FRIS-III. In the experimental section we only use FRIS-I and refer to it as FRIS.

An approach that is related to FRIS is the Positive Region based Nearest Neighbour (POSNN, [59]) classifier. In this work, the Fuzzy Nearest Neighbour (FNN, [60]) classifier of Keller is improved by weighting the instances according to their fuzzy rough positive region membership. That is to say, instead of performing a preprocessing step to the k NN classifier, a new classifier is introduced that weights instances according to their fuzzy positive region membership.

3. FRPS: Fuzzy Rough Prototype Selection

In this section we present our new PS technique: the FRPS algorithm. We stress that it is an editing technique, that is, our main goal is to improve the classification based on the k NN rule over the reduced decision system. The main outline of the algorithm is as follows:

1. Order the instances according to a measure, called alpha, inspired by fuzzy rough set theory.
2. Based on this ranking, decide which instances to remove from the training set.

In Section 3.1, we handle the first part: based on fuzzy rough set theory, we impose an order on the instances. In Section 3.2, we present the final PS algorithm that determines which threshold to use to prune the inferior instances.

3.1. A fuzzy rough measure

In this section we introduce a measure based on fuzzy rough set theory [6] to express the lack of predictive ability of instances. We first recall the main components of rough set theory and then extend it to fuzzy rough set theory.

We consider a decision system $(X,\mathcal{A}\cup\{d\})$ that consists of a universe of instances $X=\{x_1,\ldots,x_n\}$, a set of attributes $\mathcal{A}=\{a_1,\ldots,a_m\}$ and a fixed decision attribute $d\notin\mathcal{A}$. The value of instance x for attribute a is denoted by $a(x)$. The decision of instance x is denoted by $d(x)$. As we only consider classification problems, $d(x)$ takes values in a finite set of classes.

The indiscernibility relation R_{ind} is defined as

$$R_{ind} = \{(x,y)|\forall a\in\mathcal{A} : a(x) = a(y)\}. \tag{1}$$

Clearly, R_{ind} is an equivalence relation. Its equivalence classes, defined by

$$\forall x\in X : [x]_{R_{ind}} = \{y\in X|\forall a\in\mathcal{A} : a(x) = a(y)\}, \tag{2}$$

can be used to approximate concepts, i.e., subsets of the universe X. Given $A\subseteq X$, its lower and upper approximation w.r.t. R_{ind} are defined by

$$R_{ind}\downarrow A = \{x\in X|[x]_{R_{ind}}\subseteq A\} \tag{3}$$

$$R_{ind}\uparrow A = \{x\in X|[x]_{R_{ind}}\cap A\neq\emptyset\}. \tag{4}$$

Also the decision class defines an equivalence relation on X:

$$R_d = \{(x,y)|d(x) = d(y)\}. \tag{5}$$

The positive region POS is then the set containing all instances $x\in X$ for which the attributes $\mathcal{A}$ predict the decision class of x unequivocally:

$$POS = \bigcup_{x\in X} R_{ind}\downarrow[x]_{R_d}. \tag{6}$$

The more elements the POS contains, the more predictive ability the decision system has.

Classical rough set theory cannot appropriately handle continuous attributes. Therefore, in [17], rough set theory was extended using concepts from fuzzy set theory [55] to fuzzy rough set theory.

We now consider a decision system $(X,\mathcal{A}\cup\{d\})$ for which the value of instance x for attribute a can be a continuous (real) or nominal value. Without loss of generality, we assume that all continuous attributes are normalized: for all attributes $a\in\mathcal{A}$ and instances $x\in X$, $a(x)\in[0,1]$. For both the attributes in $\mathcal{A}$ and the decision class d, we construct a [0,1]-valued indiscernibility relation on the universe of instances X. The indiscernibility relation with respect to the decision class is given by:

$$\forall x,y\in X : R_d(x,y) = \begin{cases} 1 & \text{if } d(y) = d(x) \\ 0 & \text{else.} \end{cases} \tag{7}$$

The indiscernibility relation with respect to the attribute set $\mathcal{A}$ is given by

$$\forall x,y\in X,\quad \alpha\in[0,+\infty) : R^\alpha_{\mathcal{A}}(x,y) = \underbrace{T(\max(0,1-\alpha\delta_a(x,y)))}_{a\in\mathcal{A}}, \tag{8}$$

where T is a triangular norm[1] and where δ_a is a distance measure based on the attribute a. Note that this definition is valid even if $\mathcal{A}$ contains more than 2 attributes, as a t-norm is associative and can hence be extended to $[0,1]^m\rightarrow[0,1]$ mappings unequivocally. We use the following distance measure in case of a nominal attribute a:

$$\forall x,y\in X : \delta_a(x,y) = \begin{cases} 0 & \text{if } a(x) = a(y) \\ 1 & \text{else.} \end{cases} \tag{9}$$

In case a is continuous, we use

$$\forall x,y\in X : \delta_a(x,y) = (a(x)-a(y))^2. \tag{10}$$

As we assume that all continuous attributes are normalized, both distance measures return a value between 0 and 1.

The parameter $\alpha\in[0,+\infty)$ is called the *granularity* and expresses how large the differences between attribute values of instances need to be in order to distinguish between them. When α is smaller, the attribute values of the instances need to differ more in order to be able to discern between them. In the extreme case where $\alpha = 0$, all instances are indiscernible with respect to $R^0_{\mathcal{A}}$. When α is larger, small differences between the attribute values of two instances are sufficient to discern between them. This is illustrated in the following example:

Example 1. Consider two instances x,y and two continuous attributes a_1,a_2, such that

- $a_1(x) = 0.3$;
- $a_2(x) = 0.2$;
- $a_1(y) = 0.4$;
- $a_2(y) = 0.9$.

We use the minimum t-norm T_M. It follows that $R^1_{\{a_1,a_2\}}(x,y) = 0.51$, that is, it is possible to discern between the instances x and y. On the other hand, $R^{0.1}_{\{a_1,a_2\}}(x,y) = 0.951$ which means that x and y are almost indiscernible with respect to $R^{0.1}_{\{a_1,a_2\}}$.

This indiscernibility relation can be used to approximate concepts, which are now fuzzy sets $A : X\rightarrow[0,1]$. We use the definitions introduced by Radzikowska and Kerre in [61] to fuzzify the lower and upper approximation in Eqs. (3) and (4):

$$(R_{\mathcal{A}^\alpha}\downarrow A)(x) = \inf_{y\in X}\mathcal{I}(R_{\mathcal{A}^\alpha}(x,y),R_d(x,y)), \tag{11}$$

$$(R_{\mathcal{A}^\alpha}\uparrow A)(x) = \sup_{y\in X}\mathcal{T}(R_{\mathcal{A}^\alpha}(x,y),R_d(x,y)), \tag{12}$$

where inf and sup are the largest lower and smallest upper bounds of the sets respectively, and $\mathcal{I}$ is a fuzzy implicator.[2]

The fuzzy rough positive region can be used to express to what extent the attributes $\mathcal{A}$ determine the decision class of a given instance, and is defined as

$$\forall y\in X : POS^\alpha_{\mathcal{A}}(y) = \min_{x\in X}(R^\alpha_{\mathcal{A}}\downarrow,[y]_{R_d})(x), \tag{13}$$

and can be rewritten as [18]

$$\forall y\in X : POS^\alpha_{\mathcal{A}}(y) = \min_{x\in X} I(R^\alpha_{\mathcal{A}}(x,y),R_d(x,y)). \tag{14}$$

For a particular granularity α, this formula evaluates to what degree all objects indiscernible from y also belong to y's class.

We might now use the positive region to order the instances: instances with a higher positive region have more predictive ability than others. However, the problem remains to decide which granularity value α to use. So instead of using the positive region directly, we consider the following measure:

$$\forall x\in X : \alpha(x) = \sup\{\alpha\in[0,+\infty)|POS^\alpha_{\mathcal{A}}(x) < 1,\} \tag{15}$$

where $\alpha(x)$ is the minimum value α for which x fully belongs to the positive region $POS^\alpha_{\mathcal{A}}$. When for $x,y\in X$, $\alpha(x) > \alpha(y)$, it means that there are values α for which x does not fully belong to the positive region $POS^\alpha_{\mathcal{A}}$, and y does, meaning that the quality of instance y is better than that of instance x. Hence, we can use $\alpha(x)$ to measure the lack of predictive ability. It can occur that the supremum is

[1] A triangular norm (t-norm) is a mapping $T : [0,1]^2\rightarrow[0,1]$ that satisfies

- $\forall x\in[0,1] : T(x,1) = x$;
- $\forall x,y\in[0,1] : T(x,y) = T(y,x)$;
- $\forall x,y,z\in[0,1] : T(x,T(y,z)) = T(T(x,y),z)$;
- $\forall x,y,v,w\in[0,1]$: if $x\leq v$ and $y\leq w$, then $T(x,y)\leq T(v,w)$.

Examples are the Lukasiewicz t-norm T_L, the product t-norm T_P and the minimum t-norm T_M, defined for $x,y\in[0,1]$ as: $T_L(x,y) = \max(0,x+y-1), T_P(x,y) = xy$ and $T_M(x,y) = \min(x,y)$.

[2] A fuzzy implicator is a mapping $I : [0,1]^2\rightarrow[0,1]$ that satisfies

- $I(0,0) = 1$;
- $I(1,x) = x$;
- I is decreasing in the first argument;
- I is increasing in the second argument.

Examples are the Lukasiewicz implicator I_L and the Kleene Dienes implicator I_K given by, for $x,y\in[0,1] : I_L(x,y) = \min(1,1-x+y)$, $I_K(x,y) = \max(1-x,y)$.

equal to infinity, in this case, we use the symbolic notation $\alpha(x)=\infty$. This means that $\forall\alpha\in[0,+\infty)$, $POS^{\alpha}_{\mathcal{A}}(x)=0$, that is, x can never belong to the positive region to any extent.

Summarizing, instances with a low α value have higher positive regions, which means that instances indiscernible from them also belong to the same class. These instances are more *typical* for their class and have more predictive ability. Instances with a high α value have lower positive regions, which means that there are instances similar to them have different classes. These can be boundary instances, instances in overlapping regions or instances that are mislabelled. Our algorithm will remove this type of instances, and only the instances typical for their class are retained.

It is difficult to calculate the value $\alpha(x)$ directly. Therefore, we will introduce the *minimum granularity theorem* to give an expression for $\alpha(x)$ that can be used in practice.

First, we show that higher granularity parameters lead to higher fuzzy positive region membership degrees:

Lemma 1.

$$\forall y\in X,\ \forall\alpha_1,\alpha_2\in[0,+\infty):\alpha_1\le\alpha_2\Rightarrow:POS^{\alpha_1}_{\mathcal{A}}(y)\le POS^{\alpha_2}_{\mathcal{A}}(y) \tag{16}$$

Proof. Assume $y\in X$, $a\in\mathcal{A}$, $0\le\alpha_1\le\alpha_2$. Then we have

$$\forall x\in X:\max(0,1-\alpha_1\delta_a(x,y))\ge\max(0,1-\alpha_2\delta_a(x,y)). \tag{17}$$

As t-norms are increasing in both arguments, this means

$$\begin{aligned}\forall x\in X: R^{\alpha_1}_{\mathcal{A}}(x,y)&=\underbrace{T(\max(0,1-\alpha_1\delta_a(x,y)))}_{a\in\mathcal{A}}\\&\ge\underbrace{T(\max(0,1-\alpha_2\delta_a(x,y)))}_{a\in\mathcal{A}}=R^{\alpha_2}_{\mathcal{A}}(x,y).\end{aligned} \tag{18}$$

As implicators are decreasing in the first argument, this leads to:

$$POS^{\alpha_1}_{\mathcal{A}}(y)=\min_{x\in X}I(R^{\alpha_1}_{\mathcal{A}}(x,y),R_d(x,y))\le\min_{x\in X}I(R^{\alpha_2}_{\mathcal{A}}(x,y),R_d(x,y))=POS^{\alpha_2}_{\mathcal{A}}(y). \tag{19}$$

This lemma allows us to give an expression for $\alpha(x)$.

Theorem 1 (*Minimum granularity theorem*). *Let I be an implicator such that $\forall t\in[0,1]$, $I(t,0)=1-t$ holds (which is the case for e.g. the Lukasiewicz or Kleene-Dienes implicator), and let $x\in X$. Then if $T=T_M$ or $T=T_P$:*

$$\sup\{\alpha\in[0,+\infty)\Big|POS^{\alpha}_{\mathcal{A}}(x)<1\}=\max_{y\notin[x]_d}\frac{1}{\max^{m}_{i=1}\delta_{a_i}(x,y)} \tag{20}$$

and if $T=T_L$:

$$\sup\{\alpha\in[0,+\infty)\Big|POS^{\alpha}_{\mathcal{A}}(x)<1\}=\max_{y\notin[x]_d}\frac{1}{\sum^{m}_{i=1}\delta_{a_i}(x,y)}, \tag{21}$$

provided the denominators are different from zero.

Proof.

$$\begin{aligned}&POS^{\alpha}_{\mathcal{A}}(x)<1\\&\quad\Leftrightarrow\min_{y\in X}I(R^{\alpha}_{\mathcal{A}}(x,y),\ R_d(x,y))<1\\&\quad\Leftrightarrow\min_{y\notin[x]_d}I(R^{\alpha}_{\mathcal{A}}(x,y),0)<1\\&\quad\Leftrightarrow\min_{y\notin[x]_d}(1-R^{\alpha}_{\mathcal{A}}(x,y))<1\\&\quad\Leftrightarrow1-\max_{y\notin[x]_d}R^{\alpha}_{\mathcal{A}}(x,y)<1\\&\quad\Leftrightarrow\max_{y\notin[x]_d}T(\max(0,1-\alpha\delta_{a_1}(x,y)),\ldots,\max(0,1-\alpha\delta_{a_m}(x,y)))>0\end{aligned}$$

For these equivalences, we used the definition of the positive region membership function, the fact that $I(t,1)=1$ for all $t\in[0,1]$, the assumption that $I(t,0)=1-t$ for all $t\in[0,1]$ and the definition of $R^{\alpha}_{\mathcal{A}}$.

Now assume that $T=T_P$ or $T=T_M$, then

$$\begin{aligned}&POS^{\alpha}_{\mathcal{A}}(x)<1\\&\quad\Leftrightarrow\max_{y\notin[x]_d}\min_{i=1}^{m}(\max(0,1-\alpha\delta_{a_i}(x,y)))>0\\&\quad\Leftrightarrow\max_{y\notin[x]_d}\min_{i=1}^{m}(1-\alpha\delta_{a_i}(x,y))>0\\&\quad\Leftrightarrow\max_{y\notin[x]_d}(1-\alpha\max_{i=1}^{m}\delta_{a_i}(x,y))>0\\&\quad\Leftrightarrow(\exists y\notin[x]_d)\left(\alpha<\frac{1}{\max^{m}_{i=1}\delta_{a_i}(x,y)}\right)\\&\quad\Leftrightarrow\left(\alpha<\max_{y\notin[x]_d}\frac{1}{\max^{m}_{i=1}\delta_{a_i}(x,y)}\right)\end{aligned}$$

The first equivalence follows because for $T=T_M$ or $T=T_P$, $T(x,y)=0$ holds iff $x=0$ or $y=0$, for $x,y\in[0,1]$.

From these equivalences, it follows that

$$\sup\{\alpha\in[0,+\infty)\Big|POS^{\alpha}_{\mathcal{A}}(x)<1\}=\max_{y\notin[x]_d}\frac{1}{\max^{n}_{i=1}\delta_{a_i}(x,y)}. \tag{22}$$

On the other hand, when $T=T_L$, it follows that

$$\begin{aligned}&POS^{\alpha}(x)<1\\&\quad\Leftrightarrow(\exists y\notin[x]_d)(T_L(\max(0,1-\alpha\delta_{a_1}(x,y)),\ldots,\max(0,1-\alpha\delta_{a_m}(x,y)))>0)\\&\quad\Leftrightarrow(\exists y\notin[x]_d)(T_L(1-\alpha\delta_{a_1}(x,y),\ldots,1-\alpha\delta_{a_m}(x,y))>0)\\&\quad\Leftrightarrow(\exists y\notin[x]_d)(\max(0,1-\alpha\delta_{a_1}(x,y)+\cdots+1-\alpha\delta_{a_m}(x,y)-m+1)>0)\\&\quad\Leftrightarrow(\exists y\notin[x]_d)(\alpha\delta_{a_1}(x,y)+\cdots+\alpha\delta_{a_m}(x,y)<1)\\&\quad\Leftrightarrow(\exists y\notin[x]_d)\left(\alpha<\frac{1}{\sum^{m}_{i=1}\delta_{a_i}(x,y)}\right)\end{aligned}$$

In the second equivalence, we used the fact that $\forall s,t\in[0,1]$, $T_L(s,t)>0$ implies that both $s>0$ and $t>0$. Next, we used the associativity of T_L to obtain its definition for more than 2 arguments:

$$T_L(s_1,\ldots,s_m)=T_L(s_1,T_L(s_2,T_L(s_3,\ldots)))=\max(0,s_1+s_2+\cdots+s_m-m+1) \tag{23}$$

The equivalences imply

$$\sup\{\alpha\in[0,+\infty)\Big|POS^{\alpha}_{\mathcal{A}}(x)<1\}=\max_{y\notin[x]_d}\frac{1}{\sum^{m}_{i=1}\delta_{a_i}(x,y)}.\qquad\Box \tag{24}$$

A possible drawback of the measure $\alpha(x)$ is that it is max-based. This means that small changes in the data may alter the $\alpha(x)$ values drastically, and hence the robustness of the final PS method is limited. Therefore, we consider a generalization of the basic measure using Ordered Weighted Average (OWA, [30]) aggregation operators.

Recall that given a series of values $a_1,\ldots,a_p\in\mathbb{R}$ and a weight vector $W=\langle w_1,\ldots,w_p\rangle$ that fulfills:

- $\forall i\in1,\ldots,p:w_i\in[0,1]$,
- $\sum\limits_{i=1}^{p}w_i=1$,

the OWA aggregation of these values is given by

$$OWA_W(a_1,\ldots,a_p)=\sum_{i=1}^{p}w_ib_i, \tag{25}$$

where $b_i=a_j$ if a_j is the ith largest value in $a_1,\ldots,a_p$. That is, the values are ordered and then a weighted average is applied to these values.

The OWA aggregator resembles the weighted average but it assigns weights to the ordered positions of the values instead of to the values themselves. It is a very flexible aggregation operator that includes other aggregators such as minimum, maximum or average as special cases. It can also be used to relax the notion of maximum. Consider a weight vector $W=\langle w_1,\ldots,w_p\rangle$ such that $w_1\ge w_2\ge\cdots\ge w_p$. Then the operator OWA_W relaxes the maximum operator: larger values are associated with high weights, while smaller values are associated with low weights.

From now on, W_{max} is a weight vector such that $OWA_{W_{max}}$ relaxes the maximum operator. We can use this operator to calculate the $\alpha(x)$ values for each $x\in X$. For $T=T_L$, the OWA-generalized definition of $\alpha(x)$ is given by

$$\forall x\in X : \alpha^{OWA}(x)=\underbrace{OWA_{W_{max}}\frac{1}{\sum_{i=1}^{m}\delta_{a_i}(x,y)}}_{y\notin[x]_d}, \tag{26}$$

while for $T=T_M$ or $T=T_P$, we can generalize the definition of $\alpha(x)$ by

$$\forall x\in X : \alpha^{OWA}(x)=\underbrace{OWA_{W_{max}}\frac{1}{\underbrace{OWA_{W_{max}}\delta_{a_i}(x,y)}_{i=1,\ldots,m}}}_{y\notin[x]_d}. \tag{27}$$

The advantage of the OWA_W operator is that all weights can be non-zero, which means that all values can influence the aggregation result and more stable results are obtained.

A possible weight vector for an OWA aggregator with the behaviour of a maximum operator is given by:

$$\forall i\in 1,\ldots,p : w_i=\frac{2(p-i+1)}{p(p+1)}. \tag{28}$$

As a result, we have four possible definitions for the α measure, each leading to a different FRPS algorithm. In Table 1, an overview of the methods we consider is given. The first two FRPS methods use T_M or T_P as t-norm (recall from Theorem 1 that these t-norms lead to the same result), whereas the last two methods use the T_L t-norm. The weight vector W refers to the weights defined in (28).

3.2. The FRPS algorithm

Using the measure defined in the previous subsection, we can order the instances based on their quality. If we have a good threshold τ, then we can define an algorithm that removes instances $x\in X$ if $\alpha(x)>\tau$. The outline of this algorithm, which we call basic FRPS (bFRPS), is given in Algorithm 1.

Algorithm 1. bFRPS.

```
input: Decision system $(X,\mathcal{A}\cup\{d\})$, threshold $\tau$.
Calculate $\alpha(x_1),\ldots,\alpha(x_n)$
$S\leftarrow\emptyset$
for $x\in X$ do
  if $\alpha(x)\le\tau$ then
    $S\leftarrow S\cup\{x\}$
  end if
end for
Output Decision system $(S,\mathcal{A}\cup\{d\})$
```

Table 1
Overview of the $\alpha(x)$ definitions used in the FRPS methods.

Name of method	$\alpha(x)$ used
FRPS-1	$\max_{y\notin[x]_d}\frac{1}{\max_{i=1}^{m}\delta_{a_i}(x,y)}$
FRPS-2	$OWA_W\left(\frac{1}{OWA_W(\delta_{a_i}(x,y))}\right)$
FRPS-3	$\max_{y\notin[x]_d}\frac{1}{\sum_{i=1}^{m}\delta_{a_i}(x,y)}$
FRPS-4	$OWA_W\left(\frac{1}{\sum_{i=1}^{m}\delta_{a_i}(x,y)}\right)$

To determine the threshold τ, we use a wrapper approach. That is, we try several values for τ and then select the best one. To determine which is the best one, we use a leave-one-out strategy to calculate the training accuracy. The outline of this procedure, called *trainAcc* is given in Algorithm 2. To classify the instances in X, two cases are considered. If $x\in X$ is not in the selected set of prototypes S, then we assign x to the class of the nearest neighbour of x in S. In case x belongs to the set of prototypes S, we assign x to the class of the nearest neighbour of x in $S\backslash x$. If we did not make this distinction, all instances in S would be classified correctly, which would favour larger subsets of prototypes.

Algorithm 2. trainAcc, procedure to measure the training accuracy of a subset of instances using a leave-one-out approach.

```
input: Reduced decision system $(S,\mathcal{A}\cup\{d\})$ $(S\subseteq X)$.
$acc\leftarrow 0$
for $x\in X$ do
  if $x\in S$ then
    Find the nearest neighbour $nn$ of $x$ in $S\backslash\{x\}$
    if $d(x)=d(nn)$ then
      $acc\leftarrow acc+1$
    end if
  else
    Find the nearest neighbour $nn$ of $x$ in $S$
    if $d(x)=d(nn)$ then
      $acc\leftarrow acc+1$
    end if
  end if
end for
Output $acc$
```

The final question that remains is which thresholds τ to evaluate using the *trainAcc* procedure. The FRPS algorithm uses all values $\alpha(x)$ with $x\in X$ as a possible threshold. The final algorithm is given in Algorithm 3.

Algorithm 3. FRPS.

```
1:  input: Decision system $(X,\mathcal{A}\cup\{d\})$
2:  Calculate $\alpha(x_1),\ldots,\alpha(x_n)$
3:  Remove duplicates and order the $\alpha$ values from step 2:
    $\alpha_1>\alpha_2>\cdots>\alpha_p$
4:  opt.alphas$\leftarrow\{\infty\}$
5:  Calculate nearest neighbours of all instances
6:  acc.opt$\leftarrow$trainAcc$(X,\mathcal{A}\cup\{d\})$
7:  acc.current$\leftarrow$acc.opt
8:  for $\alpha=\alpha_2,\ldots,\alpha_p$ do
9:    Remove instances $x$ for which $\alpha(x)>\alpha$, the resulting set
    of instances is $S$
10:   if Number of remaining instances $>1$ then
11:     Recalculate nearest neighbours of instances for which
    current nearest neighbour was removed in step 9
12:     acc.current$\leftarrow$trainAcc$(S,\mathcal{A}\cup\{d\})$
13:     if acc.current $>$ acc.opt then
14:       opt.alphas$\leftarrow\{\alpha\}$
15:     else if acc.current $=$ acc.opt then
16:       opt.alphas$\leftarrow$opt.alphas$\cup\{\alpha\}$
17:     end if
18:   end if
19: end for
20: best.alpha$=$median(opt.alphas)
21: Output bFRPS$((X,\mathcal{A}\cup\{d\}),best.alpha)$
```

First, the $\alpha(x)$ values are calculated for each $x\in X$. Then, duplicates are removed from these values and they are ordered.

In line 5, the nearest neighbour is calculated for each instance. Next, in line 6 the accuracy using the entire instance set is calculated. In each run of the loop from line 8 to line 19 instances x for which $\alpha(x)$ exceeds the current threshold α are removed from the current instance set. If the accuracy of the current instance subset is equal or better than the best accuracy reached so far, the best accuracy and the corresponding list of optimal alphas are updated. Finally, the best α value is calculated as the median[3] of all optimal α values and used as threshold.

The reason why we consider the α values in decreasing order is that we can implement this efficiently. The instance subsets are generated decrementally. Every time instances are removed from the current instance subset, we only recalculate the nearest neighbours of those instances for which the nearest neighbour is removed in the current step.

The stopping criterion in line 10 makes sure that a nearest neighbour can be calculated for every instance: if there is only one instance x for which $\alpha(x) = \alpha_p$, that is, there is only one instance with the lowest α value, then in line 11 the nearest neighbour cannot be calculated for x: the only candidate instance is x but this instance cannot be picked as nearest neighbour of x in the leave-one-out strategy. Therefore, we choose not to consider this instance subset $\{x\}$.

The next toy example shows how the FRPS procedure works.

Example 2. Consider a decision system with 2 attributes a_1 and a_2 and two decision classes 0 and 1. There are 6 instances $\{x_1,\ldots,x_6\}$. The decision system is shown in Table 2. The first step of FRPS is to calculate the $\alpha(x)$ values for each instance x. We use the $\alpha(x)$ definition where the T_L t-norm is used:

$$\alpha(x_1) = \max\left(\frac{1}{0.04+0.01}, \frac{1}{0.04+0.49}, \frac{1}{0.09+0.09}\right) = 20$$
$$\alpha(x_2) = \max\left(\frac{1}{0.01+0.04}, \frac{1}{0.25+0.16}, \frac{1}{0+0}\right) = \infty$$
$$\alpha(x_3) = \max\left(\frac{1}{0.04+0.01}, \frac{1}{0.01+0.04}, \frac{1}{0.01+0.04}\right) = 20$$
$$\alpha(x_4) = \max\left(\frac{1}{0.04+0.49}, \frac{1}{0.25+0.16}, \frac{1}{0.25+0.16}\right) = 2.41$$
$$\alpha(x_5) = \max\left(\frac{1}{0.01+0.04}, \frac{1}{0.25+0.16}, \frac{1}{0+0}\right) = \infty$$
$$\alpha(x_6) = \max\left(\frac{1}{0.09+0.09}, \frac{1}{0+0}, \frac{1}{0+0}\right) = \infty$$

The FRPS algorithm orders the distinct α values from high to low:

$$\infty > 20 > 2.41.$$

First, we consider the entire instance set $\{x_1,x_2,x_3,x_4,x_5,x_6\}$. The nearest neighbours of each instance are given in Table 3. Note that if an instance has more than 1 nearest neighbour, one of them is picked at random. Applying 1NN now misclassifies x_2, x_4, x_5, x_6 and classifies x_1 and x_3 correctly. This means that the training accuracy is now $\frac{2}{6}$. In the next step, we consider $\alpha = 20$. Now, x_2,x_5 and x_6 are removed. We only have to recalculate the nearest neighbours of x_2, x_4, x_5 and x_6. They are given in Table 3. Now, x_1, x_2, x_3 are classified correctly by $1NN$ and x_4, x_5, x_6 incorrectly. The training accuracy is now $\frac{3}{6}$.

Next, we consider $\alpha = 2.41$. Now, only x_4 remains in the dataset, so the procedure stops. We conclude that using $\alpha = 20$ yields the best training accuracy, and therefore, we return $\{x_1,x_3,x_4\}$ as prototypes.

Table 2
Decision system Example 2.

	a_1	a_2	d
x_1	0.5	0.1	0
x_2	0.2	0.4	0
x_3	0.3	0.2	0
x_4	0.7	0.8	1
x_5	0.2	0.4	1
x_6	0.2	0.4	1

Table 3
Nearest neighbours of the instances at each step in Example 2.

	$\alpha = \infty$	$\alpha = 20$
x_1	x_3	x_3
x_2	x_5	x_3
x_3	x_1	x_1
x_4	x_2	x_3
x_5	x_2	x_3
x_6	x_2	x_3

If we assume that the number of neighbours in k NN is a small constant (in our experiments k=1 or k=3), the complexity of *FRPS* is $\mathcal{O}(n^3m)$. Line 2 in Algorithm 3 requires $\mathcal{O}(n^2m)$ calculations. The most costly step, however, is the loop in lines 8–20: for each instance, a subset is generated for which the classification accuracy is calculated. Therefore, the NN rule needs to be performed for each instance. Each NN evaluation requires the calculation of the distances to each instance in the subset of selected instances, i.e., its cost is at most $O(n^2m)$. In practice, the complexity will be lower, as only the nearest neighbours of those instances for which the nearest neighbour is removed need to be recalculated.

4. Experimental evaluation

In this section we evaluate the performance of the FRPS algorithm and compare it to a range of state-of-the-art PS algorithms. In Section 4.1, we describe the experimental set-up of our evaluation, and in Section 4.2 we present the results.

4.1. Experimental set-up

To show the good performance of the FRPS algorithm, we follow the experimental set-up as described in [31]. We consider 58 datasets and their partitions from the KEEL dataset repository[4] [62]. The main characteristics of these datasets are given in Table 4. We consider several types of datasets: the datasets contain from 100 up to 19,000 instances and the number of attributes ranges from 2 to 85. Some of the datasets contain only continuous attributes (e.g. appendicitis), others contain only nominal attributes (e.g. breast) and the others contain both (e.g. abalone). Some of these datasets originally contained instances for which attribute values were missing. We removed these instances from the datasets, the numbers in Table 4 correspond to the datasets without missing data.

Guided by the results in [31], we select 22 Prototype Selection methods against which we compare the FRPS algorithm. An overview of these methods is given in Table 5. They are representative in the sense that they are the best performing methods among each type of methods discussed in [31]. We also run the

[3] We opt to take the median because it is a compromise between removing possibly useful instances and retaining too many instances. For completeness, we also added results of the FRPS algorithm using the minimum or maximum of the optimal α values on our web site: http://users.ugent.be/~nverbies/.

[4] http://www.keel.es/datasets.php.

Table 4
Characteristics of the datasets used in the experimentation: number of instances (# Inst.), number of attributes (# Atts.), number of continuous (# Cont.) and nominal (# Nom.) attributes, number of classes (# Cl.).

Dataset	# Inst.	# Atts.	# Cont.	# Nom.	# Cl.
abalone	4174	8	7	1	28
appendicitis	106	7	7	0	2
australian	690	14	8	6	2
automobile	205	25	15	10	6
balance	625	4	4	0	3
banana	5300	2	2	0	2
bands	539	19	19	0	2
breast	286	9	0	9	2
bupa	345	6	6	9	2
car	1728	6	0	6	4
chess	3196	36	0	36	2
cleveland	303	13	13	0	5
coil2000	9822	85	85	0	2
contraceptive	1473	9	9	0	3
crx	690	15	6	9	2
dermatology	366	34	34	0	6
ecoli	336	7	7	0	8
flare-solar	1066	11	0	11	6
german	1000	20	7	13	2
glass	214	9	9	0	7
haberman	306	3	3	0	2
hayes-roth	160	4	4	0	3
heart	270	13	13	0	2
hepatitis	155	19	19	0	2
housevotes	435	16	0	16	2
iris	150	4	4	0	3
led7digit	500	7	7	0	10
lymphography	148	18	3	15	4
magic	19,020	10	10	0	2
mammographic	961	5	5	0	2
marketing	8993	13	13	0	9
monk-2	432	6	6	0	2
newthyroid	215	5	5	0	3
nursery	12,960	8	0	8	5
pageblocks	5472	10	10	0	5
penbased	10,992	16	16	0	10
phoneme	5404	5	5	0	2
pima	768	8	8	0	2
ring	7400	20	20	0	2
saheart	462	9	8	1	2
satimage	6435	36	36	0	7
segment	2310	19	19	0	7
sonar	208	60	60	0	2
spambase	4597	57	57	0	2
spectheart	267	44	44	0	2
splice	3190	60	0	60	3
tae	151	5	5	0	3
texture	5500	40	40	0	11
thyroid	7200	21	21	0	3
tic-tac-toe	958	9	0	9	2
titanic	2201	3	3	0	2
twonorm	7400	20	20	0	2
vehicle	846	18	18	0	4
vowel	990	13	13	0	11
wine	178	13	13	0	3
wisconsin	699	9	9	0	2
yeast	1484	8	8	0	10
zoo	101	16	0	16	7

FRIS algorithm, with threshold τ set to 1 and $\alpha = 10$ in the indiscernibility relation.[5]

We follow a 10-fold cross-validation strategy to evaluate the algorithms; in each fold the data is divided into a training and testing part. We use FRPS to reduce the training data and apply kNN on the test data, looking up the nearest neighbours in the reduced training set. Four evaluation parameters are considered:

- *Test Accuracy* (*acc*): the rate of correctly classified instances in the test data.
- *Test Cohen's kappa* (κ) [63]: this is an additional accuracy measure that compensates for random hits. Given the confusion matrix $[y_{ij}]_{\Omega \times \Omega}$ of the classification task (Ω is the number of classes) it is given by:

$$\kappa = \frac{n \sum_{i=1}^{\Omega} y_{ii} - \sum_{i=1}^{\Omega} y_{i.} y_{.i}}{n^2 - \sum_{i=1}^{\Omega} y_{i.} y_{.i}}, \tag{29}$$

  where $\forall i = 1, \ldots, \Omega$, $y_{.i}$ and $y_{i.}$ are the sum of the elements of the ith column and ith row of the confusion matrix respectively.
- *Storage reduction* (*red*): the fraction of instances removed from the training data.
- *Running time* (*time*): this is the running time in seconds of the Prototype Selection method. The running time of the subsequent 1NN classification is not taken into account.

We also perform statistical comparisons over the multiple datasets considered to find significant differences between FRPS and the remaining methods. In [64], it is recommended to use a set of simple, safe and robust non-parametric tests for statistical comparisons of classifiers.

We apply the Wilcoxon's signed ranks statistical test [65] to compare FRPS against all other considered PS methods. This is a non-parametric pairwise test that aims to detect significant differences between two sample means; that is, the behaviour of the two implicated algorithms in the comparison. For each comparison we compute the sum of ranks of Wilcoxon's test in favour of FRPS $R+$, the sum of ranks in favour of the other methods $R-$ and also the p-value obtained for the comparison.

Besides, we perform a statistical analysis conducted by non-parametric multiple comparison procedures [66–68]. We use Friedman's procedure to compute the set of ranks that represent the effectiveness associated with each algorithm. We compute the p-value related to the significance of the differences found by this test. In addition, we compute the adjusted p-value with Holm's test. More information about these tests and other statistical procedures can be found at http://sci2s.ugr.es/sicidm/.

We take the results from [31] for the state-of-the-art methods. The FRPS method is implemented within the KEEL software platform, and the experiments are run on the same machine as the state-of-the-art methods.

4.2. Results

In this section we present the results of our approach. Due to space restrictions, we are not able to include all the details in this paper, but they can be found on our webpage.[6] In Section 4.2.1 we use the 1NN classifier to evaluate FRPS, while in Section 4.2.2 we use FRPS as a preprocessing method for the 3NN classifier to study how FRPS performs for higher values of k.

4.2.1. Results using 1NN as classifier

Before comparing our approach to the state-of-the-art algorithms, we evaluate the performance of the different $\alpha(x)$ definitions in Table 1. We use the non-parametric statistical Wilcoxon test to compare each of the FRPS algorithms to each other. In Table 6, we show the average results of the FRPS methods on all datasets. Recall that we use four versions of the FRPS algorithm,

[5] Results with α ranging from 1 to 10 can be found on or web site http://users.ugent.be/~nverbies/.

[6] http://users.ugent.be/~nverbies/

Table 5
State-of-the-art PS algorithms against which FRPS is compared. The full name, abbreviation and reference are given, as well as its place in the PS taxonomy.

Complete name	Abbrev. name	Reference	Type of method
All-k NN	All$k$ NN	[34]	Editing—Filter
Class Conditional Instance Selection	CCIS	[51]	Hybrid—Filter
CHC Evolutionary Algorithm	CHC	[48]	Hybrid—Wrapper
Condensed Nearest Neighbour	CNN	[37]	Condensation—Filter
C-Pruner	Cpruner	[52]	Hybrid—Filter
Decremental Reduction Optimization Procedure 3	DROP3	[50]	Hybrid—Filter
Fast Condensed Nearest Neighbour 1	FCNN	[39]	Condensation—Filter
Generational Genetic Algorithm	GGA	[44]	Hybrid—Wrapper
Hit Miss Network Edition Iterative	HMNEI	[49]	Hybrid—Filter
Instance Based 3	IB3	[53]	Hybrid—Filter
Iterative Case Filtering	ICF	[54]	Hybrid—Filter
Modified Condensed Nearest Neighbour	MCNN	[40]	Condensation—Filter
Modified Edited Nearest Neighbour	MENN	[33]	Editing—Filter
Model Class Selection	MoCS	[36]	Editing—Filter
Modified Selective Subset	MSS	[42]	Condensation—Filter
Patterns by Ordered Projections	POP	[41]	Condensation— Filter
Reconsistent	Reconsistent	[43]	Condensation—Filter
Random Mutation Hill Climbing	RMHC	[46]	Hybrid —Wrapper
Relative Neighbourhood Graph Editing	RNG	[35]	Editing—Filter
Reduced Nearest Neighbour	RNN	[38]	Condensation—Filter
Steady-State Memetic Algorithm	SSMA	[47]	Hybrid—Wrapper
Fuzzy Rough Instance Selection	FRIS	[29]	Editing—Filter

Table 6
Average results of the four FRPS methods in Table 1, using 1NN.

Method	acc	Kappa	Time	Red
FRPS-1	0.7804	0.5713	20.219	0.2972
FRPS-2	0.7893	0.5898	37.273	0.3568
FRPS-3	0.7868	0.5824	21.571	0.3915
FRPS-4	0.7920	0.5942	27.638	0.3557

each version corresponding to a different α measure describing the lack of predictive ability of the instances:

- *FRPS*-1 uses the T_M or T_P t-norm to aggregate the similarities of the separate attributes.
- *FRPS*-2 uses the T_M or T_P t-norm to aggregate the similarities of the separate attributes and uses an *OWA* aggregator to generalize the maximum.
- *FRPS*-3 uses the T_L t-norm to aggregate the similarities of the separate attributes.
- *FRPS*-4 uses the T_L t-norm to aggregate the similarities of the separate attributes and uses an *OWA* aggregator to generalize the maximum.

The results of the Wilcoxon test are shown in Table 7. A κ sign means that the algorithm in the row outperforms the algorithm in the column at the 10% significance level with respect to Cohen's kappa, an *acc* sign means that the algorithm in the row outperforms the algorithm in the column with respect to accuracy. An empty table cell indicates that the algorithm in the row and column do not significantly differ from each other, in this case the p-value is higher than 0.10. From these tables, we can draw two conclusions:

1. The OWA approach is beneficial: FRPS-2 performs better than FRPS-1 and FRPS-3 with respect to both accuracy and Cohen's kappa.
2. The running time of the approaches using T_L as a t-norm is lower than those using the T_P or T_M t-norm.

Based on these conclusions, we decide to use FRPS-4 and we will refer to this method as FRPS in the remainder of this work.

Table 7
Comparison between the four FRPS algorithms in Table 1 with respect to test accuracy and Cohen's kappa. An *acc* sign (resp. κ) means that the method in the row outperforms the method in the column with respect to classification accuracy (resp. Cohen's kappa). Results are obtained with 1NN.

Method	FRPS-1	FRPS-2	FRPS-3	FRPS-4
FRPS-1				
FRPS-2	κ *acc*		κ	
FRPS-3				
FRPS-4	κ *acc*			

Next, we compare the FRPS algorithm to each of the state-of-the-art algorithms in Table 5. In Table 8, we show the average results of all methods, ordered according to their performance. The algorithms that obtain the best behaviour with respect to both reduction and accuracy are the hybrid techniques RMHC, CHC and SSMA. However, the significant improvement in the accuracy rate these methods achieve comes with a high computation cost. The methods that are less accurate but that show a great reduction of time complexity are DROP3 and CCIS. If the objective is the accuracy rate, the best results are achieved with FRPS and RNG as the editing method and with HMNEI as a hybrid method. When the key factor is reduction, FCNN is the highlighted one, being one of the fastest condensation methods. The reduction rate of FRPS is, as expected, not high, and is similar to that of the other editing PS algorithms. The running time of FRPS is average: FRPS is much faster than the other wrapper methods, but slower than some of the filter methods. This is remarkable; FRPS is a wrapper method, but the decremental nature of FRPS allows one to implement it efficiently, which results in a relatively low running time.

In order to show the statistical significance of the good performance of FRPS with respect to test accuracy and Cohen's kappa, we perform the Wilcoxon test to compare FRPS to each of the state-of-the-art PS algorithms. The statistics of this test are given in Table 9. Both the $R+$ and $R-$ values are given, as well as the asymptotic p-values. For Cohen's kappa, the asymptotic p-value is lower than 0.10 for all comparisons, so we can state that FRPS significantly outperforms all state-of-the-art algorithms with respect to Cohen's kappa. For the accuracy, only the p-value for the comparison with RNG is larger than 0.10, so FRPS significantly outperforms all state-of-the-art algorithms apart from RNG with respect to accuracy.

Table 8
Average results of FRPS and the state-of-the-art PS algorithms, using 1NN.

Test acc.		Kappa		Red		Time	
FRPS	**0.7920**	**FRPS**	**0.5942**	CHC	0.9785	POP	0.0620
SSMA	0.7819	SSMA	0.5743	SSMA	0.9539	CNN	0.3626
CHC	0.7799	RMHC	0.5724	MCNN	0.9352	FCNN	1.1057
RNG	0.7798	HMNEI	0.5700	GGA	0.9302	MCNN	1.4932
RMHC	0.7792	CHC	0.5669	RNN	0.9289	IB3	2.1922
GGA	0.7762	RNG	0.5662	CCIS	0.9202	MSS	2.6236
ModelCS	0.7740	GGA	0.5632	CPruner	0.9075	FRIS	2.7999
HMNEI	0.7701	ModelCS	0.5632	RMHC	0.9011	CCIS	4.1368
AllKNN	0.7678	FRIS	0.5450	DROP3	0.8462	ModelCS	5.1000
FRIS	0.7590	AllKNN	0.5421	ICF	0.7509	AllKNN	8.1249
POP	0.7576	POP	0.5376	IB3	0.7190	HMNEI	9.5758
MENN	0.7541	MSS	0.5250	FCNN	0.6639	CPruner	11.6989
RNN	0.7520	MENN	0.5217	CNN	0.6177	MENN	12.2071
MSS	0.7497	FCNN	0.5139	Reconsistent	0.5878	ICF	30.5873
FCNN	0.7391	IB3	0.5125	HMNEI	0.5428	**FRPS**	**32.8760**
IB3	0.7389	CNN	0.5122	MSS	0.4739	DROP3	52.8063
CNN	0.7381	RNN	0.5121	MENN	0.4519	Reconsistent	534.3105
DROP3	0.7142	Reconsistent	0.4714	**FRPS**	**0.3557**	RNG	616.3506
Reconsistent	0.7124	DROP3	0.4686	AllKNN	0.3174	SSMA	2084.4480
CPruner	0.6991	MCNN	0.4410	RNG	0.2080	CHC	2244.7833
CCIS	0.6979	CCIS	0.4396	FRIS	0.1265	RMHC	3962.0403
MCNN	0.6842	ICF	0.4209	ModelCS	0.1109	GGA	7022.4551
ICF	0.6824	CPruner	0.3918	POP	0.0732	RNN	8030.2003

Table 9
Comparison of the state-of-the-art algorithms with FRPS, with respect to accuracy and Cohen's kappa, using 1NN.

FRPS vs	acc			κ		
Method	$R+$	$R-$	Asymptotic p-value	$R+$	$R-$	Asymptotic p-value
AllKNN	1283.5	369.5	0.000267	1334.0	319.0	0.000053
CCIS	1583.5	69.5	0	1613.5	97.5	0
CHC	1073.0	580.0	0.049482	1126.0	527.0	0.017146
CNN	1615.0	96.0	0	1465.0	246.0	0.000002
CPruner	1588.5	64.5	0	1619.0	34.0	0
DROP3	1706.0	5.0	0	1641.0	70.0	0
FCNN	1588.0	123.0	0	1452.0	259.0	0.000004
GGA	1219.0	434.0	0.001775	1252.0	459.0	0.002114
HMNEI	1265.0	388.0	0.000487	1159.0	552.0	0.018475
IB3	1613.0	98.0	0	1500.0	211.0	0.000001
ICF	1708.0	3.0	0	1696.0	15.0	0
MCNN	1653.0	58.0	0	1587.0	124.0	0
MENN	1330.5	380.5	0.000232	1388.0	323.0	0.000037
ModelCS	1345.5	365.5	0.00014	1157.5	495.5	0.008378
MSS	1562.5	90.5	0	1435.5	275.5	0.000007
POP	1483.5	169.5	0	1278.5	374.5	0.000324
Reconsistent	1703.0	8.0	0	1616.0	95.0	0
RMHC	1136.0	575.0	0.029428	1139.0	572.0	0.02789
RNG	1007.0	704.0	0.239261	1104.0	607.0	0.053874
RNN	1581.0	130.0	0	1508.0	203.0	0
SSMA	1081.0	572.0	0.042762	1137.0	574.0	0.029011
1NN	1508.5	144.5	0	1363.0	290.0	0.00002
FRIS	1437.0	274.0	0.000007	1357.5	353.5	0.0001

However, when using the Wilcoxon test for multiple pairwise comparisons, we lose control on the family-wise error rate, this is the probability of making one or more false discoveries among all the hypotheses [66]. Therefore, we also use the Friedman test and Holm post hoc procedure, specifically designed for comparing multiple algorithms, to show the good performance of FRPS.

We perform the Friedman test and Holm post hoc procedure only for the nine best scoring algorithms with respect to test accuracy and Cohen's kappa, as these procedures may lose power if one compares too many algorithms [66]. The Friedman rankings are given in Table 10. For both accuracy and Cohen's kappa, FRPS obtains the best ranking. Next, we perform the Holm post hoc procedure to compare FRPS with each of the other PS algorithms.

Table 10
Results of the Friedman test: the Friedman rankings are given for both the comparison with respect to accuracy and Cohen's kappa, using 1NN.

Algorithm	Ranking w.r.t acc.	Ranking w.r.t κ
AllKNN	5.7328	6.0776
CHC	4.8879	5.4655
GGA	5.6983	5.9224
HMNEI	5.3017	4.5086
ModelCS	5.3448	5.0862
RMHC	5.4224	5.1897
RNG	3.8707	4.3966
SSMA	4.9914	4.8448
FRPS	3.75	3.5086

Table 11
Adjusted *p*-values of the Holm post hoc procedure. The FRPS method is compared against each of the algorithms in the first column and the corresponding *p*-value is given in column 2 for the comparison with respect to accuracy and in column 3 with respect to Cohen's kappa. Results are obtained using the 1NN classifier.

FRPS VS	p_{holm} acc	$p_{holm}\kappa$
AllKNN	0.000773	0.000004
GGA	0.000893	0.000014
RMHC	0.006041	0.004739
ModelCS	0.008562	0.007685
HMNEI	0.009115	0.098509
SSMA	0.043936	0.025805
CHC	0.050493	0.000714
RNG	0.812407	0.098509

Table 12
Average results of FRPS and the state-of-the-art PS algorithms, using 3NN.

Test acc.		Kappa	
FRPS	**0.784357**	**FRPS**	**0.583008**
ModelCS	0.783636	ModelCS	0.579932
RNG	0.78257	HMNEI	0.568039
RMHC	0.772238	RNG	0.567519
HMNEI	0.769689	POP	0.555516
POP	0.769608	RMHC	0.55417
AllKNN	0.769205	FRIS	0.545045
GGA	0.764862	AllKNN	0.539715
SSMA	0.764677	GGA	0.539556
FRIS	0.758969	SSMA	0.537683
MSS	0.752465	MSS	0.529244
MENN	0.750534	CNN	0.521819
CHC	0.746067	FCNN	0.518912
RNN	0.745975	MENN	0.504962
CNN	0.745906	CHC	0.497511
FCNN	0.744242	Reconsistent	0.495426
Reconsistent	0.729242	RNN	0.495111
IB3	0.717835	IB3	0.468368
ICF	0.703723	DROP3	0.443813
DROP3	0.69675	ICF	0.443687
CPruner	0.689395	CCIS	0.371281
CCIS	0.663074	CPruner	0.370772
MCNN	0.640596	MCNN	0.366144

The adjusted *p*-values are given in Table 11. We see that all algorithms except for RNG are significantly worse than FRPS with respect to test accuracy. For Cohen's kappa, FRPS clearly performs better than all other considered PS algorithms.

4.2.2. Results using 3NN as a classifier

In this section we discuss the performance of FRPS applied as preprocessing method for the 3NN classifier. In Table 12 we show the average results over all datasets. We do not list the reduction rate and execution time because they are the same as for 1NN. On average, FRPS is the best performing PS method with respect to accuracy and Cohen's kappa. To test the significance of this result, we performed the Wilcoxon test to compare FRPS with the other methods. The values of the statistics are given in Table 13. None of the considered PS methods outperforms FRPS and FRPS outperforms most of the other PS methods with respect to accuracy and Cohen's kappa. Only MoCS, RMHC and RNG are not significantly worse than FRPS with respect to both accuracy and Cohen's kappa.

From this we conclude that FRPS works not as well for 3NN as for 1NN, but it is still a good idea to use FRPS for 3NN: it is a fast method and none of the other PS methods outperforms it with respect to accuracy or Cohen's kappa.

5. Conclusion and future work

In this paper we have presented a new Prototype Selection method, FRPS. This preprocessing method is designed to only retain instances with good predictive ability and aims to improve *k* NN classification. We have done this by extending the existing FRIS method by building a wrapper around it. In order to keep the running time of this wrapper under control and to obtain a good accuracy, we have introduced the minimal granularity theorem. The OWA operator is used to refine the final algorithm. An experimental study that compares FRPS to 22 state-of-the-art Prototype Selection algorithms on 58 datasets shows its good performance.

Table 13
Comparison of the state-of-the-art algorithms with FRPS, with respect to accuracy and Cohen's kappa, using 3NN.

FRPS vs	acc			κ		
	$R+$	$R-$	Asymptotic P-value	$R+$	$R-$	Asymptotic P-value
AllKNN	1099.0	612.0	0.058385	1221.5	489.5	0.004507
CCIS	1549.0	104.0	0	1576.0	77.0	0
CHC	1217.5	493.5	0.005007	1266.0	387.0	0.000472
CNN	1530.0	181.0	0	1416.0	295.0	0.000014
CPruner	1601.0	52.0	0	1641.0	12.0	0
DROP3	1598.0	113.0	0	1551.0	160.0	0
FCNN	1498.0	155.0	0	1390.0	263.0	0.000007
GGA	1090.0	621.0	0.068306	1182.5	528.5	0.011146
HMNEI	1141.5	569.5	0.026396	1010.0	701.0	0.230115
IB3	1609.5	101.5	0	1568.0	143.0	0
ICF	1537.0	174.0	0	1569.0	142.0	0
MCNN	1678.0	33.0	0	1657.0	54.0	0
MENN	1202.0	509.0	0.007162	1367.5	343.5	0.000072
ModelCS	979.0	732.0	0.336562	968.0	685.0	0.259227
MSS	1362.0	291.0	0.000019	1333.0	378.0	0.000215
POP	1193.0	460.0	0.003547	1152.0	559.0	0.021478
Reconsistent	1529.5	181.5	0	1462.0	249.0	0.000003
RMHC	957.0	696.0	0.297958	1047.0	664.0	0.137139
RNG	812.0	841.0	1	988.0	723.0	0.303136
RNN	1373.0	280.0	0.000014	1527.0	184.0	0
SSMA	1108.0	545.0	0.024911	1182.0	471.0	0.004677
3NN	1203.5	507.5	0.006755	1167.0	544.0	0.015709
FRIS	1385.0	326.0	0.000041	1289.0	422.0	0.00077

In order to improve the reduction rate of the FRPS algorithm, we plan to combine it with condensation methods in order to develop a hybrid Fuzzy Rough Prototype Selection algorithm.

Furthermore, as k NN is not only susceptible to noise on the instance level, we want to combine FRPS with feature selection. As FRPS is a wrapper method, it uses the 1NN method and hence it might suffer from low quality features as well. This means that first performing FRPS and then feature selection does not solve the problem. On the other hand, many feature selection methods are susceptible to noisy data, which means that first performing feature selection and then FRPS is no solution either. Therefore, we want to develop an algorithm that simultaneously performs feature and instance selection in the future.

Conflict of interest statement

None declared.

Acknowledgments

The paper has been partially supported by the Spanish projects TIN2011-28488, TIC-2010-06858 and P11-TIC-7765.

References

[1] T. Cover, P. Hart, Nearest neighbor pattern classification, IEEE Transactions on Information Theory 13 (1) (1967) 21–27.

[2] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, Wiley, 2001.

[3] M.E. Farrell, A. Passamante, T. Hediger, Comparing a nearest-neighbor estimator of local attractor dimensions for noisy data to the correlation dimension, Physical Review A (Atomic, Molecular, and Optical Physics) 41 (12) (1990) 6591–6595.

[4] R.P.W. Duin, P. Paclík, Prototype selection for dissimilarity-based classifiers, Pattern Recognition 39 (2006) 189–208.

[5] M. Grochowski, N. Jankowski, Comparison of instance selection algorithms. II. Results and comments, in: Proceedings of the Seventh International Conference on Artificial Intelligence and Soft Computing, vol. 3070, 2004, pp. 580–585.

[6] Z. Pawlak, Rough sets, International Journal of Computer Information Science 11 (1982) 341–356.

[7] J.G. Bazan, H.S. Nguyen, S.H. Nguyen, P. Synak, J. Wróblewski, Rough set algorithms in classification problem, in: Rough Set Methods and Applications, 2000, pp. 49–88.

[8] C. Degang, W. Changzhong, H. Qinghua, A new approach to attribute reduction of consistent and inconsistent covering decision systems with covering rough sets, Information Science 177 (17) (2007) 3500–3518.

[9] X. Yang, J. Yang, C. Wu, D. Yu, Dominance-based rough set approach and knowledge reductions in incomplete ordered information system, Information Science 178 (4) (2008) 1219–1234.

[10] Y. Zhao, Y. Yao, F. Luo, Data analysis based on discernibility and indiscernibility, Information Sciences 177 (22) (2007) 4959–4976.

[11] A. Chouchoulas, Q. Shen, Rough set-aided keyword reduction for text categorisation, Applied Artificial Intelligence 15 (9) (2001) 843–873.

[12] R.W. Swiniarski, A. Skowron, Rough set methods in feature selection and recognition, Pattern Recognition Letters 24 (6) (2003) 833–849.

[13] F. Min, W. Zhu, Attribute reduction of data with error ranges and test costs, Information Sciences 211 (2012) 48–67.

[14] Z. Meng, Z. Shi, Extended rough set-based attribute reduction in inconsistent incomplete decision systems, Information Sciences 204 (2012) 44–69.

[15] J. Liang, F. Wang, C. Dang, Y. Qian, An efficient rough feature selection algorithm with a multi-granulation view, International Journal of Approximate Reasoning 53 (6) (2012) 867–926.

[16] D. Tian, X. Jun Zeng, J. Keane, Core-generating approximate minimum entropy discretization for rough set feature selection in pattern classification, International Journal of Approximate Reasoning 52 (6) (2011) 659–914.

[17] D. Dubois, H. Prade, Rough fuzzy sets and fuzzy rough sets, International Journal of General Systems 17 (1990) 191–209.

[18] C. Cornelis, R. Jensen, G. Hurtado, D. Slezak, Attribute selection with fuzzy decision reducts, Information Sciences 180 (2) (2010) 209–224.

[19] D. Chen, E. Tsang, E. Zhao, An approach of attributes reduction based on fuzzy rough sets, in: Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, 2007, pp. 486–491.

[20] D. Chen, E. Tsang, S. Zhao, Attribute reduction based on fuzzy rough sets, in: Proceedings of the International Conference on Rough Sets and Intelligent Systems Paradigms, 2007, pp. 83–89.

[21] Q. Hu, X. Xie, D. Yu, Hybrid attribute reduction based on a novel fuzzy-rough model and information granulation, Pattern Recognition 40 (12) (2007) 3509–3521.

[22] R. Jensen, Q. Shen, Fuzzy-rough attribute reduction with application to web categorization, Fuzzy Sets and Systems 141 (3) (2004) 469–485.

[23] R. Jensen, Q. Shen, Fuzzy-rough sets assisted attribute selection, IEEE Transactions on Fuzzy Systems 15 (1) (2007) 73–89.

[24] R. Jensen, Q. Shen, New approaches to fuzzy-rough feature selection, IEEE Transactions on Fuzzy Systems 17 (4) (2009) 824–838.

[25] E. Tsang, D. Chen, D. Yeung, X. Wang, J. Lee, Attributes reduction using fuzzy rough sets, IEEE Transactions on Fuzzy Systems 16 (5) (2008) 1130–1141.

[26] S. Zhao, E. Tsang, On fuzzy approximation operators in attribute reduction with fuzzy rough sets, Information Sciences 178 (16) (2007) 3163–3176.

[27] D. Chen, L. Zhang, S. Zhao, Q. Hu, P. Zhu, A novel algorithm for finding reducts with fuzzy rough sets, IEEE Transactions on Fuzzy Systems 20 (2) (2012) 385–389.

[28] D. Chen, Q. Hu, Y. Yang, Parameterized attribute reduction with gaussian kernel based fuzzy rough sets, Information Sciences 181 (23) (2011) 5169–5179.

[29] R. Jensen, C. Cornelis, Fuzzy-rough instance selection, in: Proceedings of the 19th International Conference on Fuzzy Systems, 2010, pp. 1776–1782.

[30] R.R. Yager, On ordered weighted averaging aggregation operators in multicriteria decision making, IEEE Transactions on Systems, Man and Cybernetics 18 (1988) 183–190.

[31] S. García, J. Derrac, J. Cano, F. Herrera, Prototype selection for nearest neighbor classification: taxonomy and empirical study, IEEE Transactions on Pattern Analysis and Machine Intelligence 34 (3) (2012) 417–435.

[32] D.L. Wilson, Asymptotic properties of nearest neighbor rules using edited data, IEEE Transactions on Systems, Man and Cybernetics 2 (3) (1972) 408–421.

[33] K. Hattori, M. Takahashi, A new edited k-nearest neighbor rule in the pattern classification problem, Pattern Recognition 32 (2000) 521–528.

[34] I. Tomek, An experiment with the edited nearest-neighbor rule, IEEE Transactions on Systems, Man and Cybernetics 6 (6) (1976) 448–452.

[35] J.S. Sánchez, F. Pla, F.J. Ferri, Prototype selection for the nearest neighbour rule through proximity graphs, Pattern Recognition Letters 18 (1997) 507–513.

[36] C.E. Brodley, Recursive automatic bias selection for classifier construction, Machine Learning 20 (1995) 63–94.

[37] P.E. Hart, The condensed nearest neighbor rule, IEEE Transactions on Information Theory 18 (1968) 515–516.

[38] G. Gates, The reduced nearest neighbor rule, IEEE Transactions on Information Theory 18 (3) (1972) 431–433.

[39] F. Angiulli, Fast nearest neighbor condensation for large data sets classification, IEEE Transactions on Knowledge and Data Engineering 19 (11) (2007) 1450–1464.

[40] V.S. Devi, M.N. Murty, An incremental prototype set building technique, Pattern Recognition 35 (2) (2002) 505–513.

[41] J. Riquelme, J. Aguilar-Ruiz, J.S. Aguilar-ruiz, M. Toro, Finding representative patterns with ordered projections, Pattern Recognition 36 (4) (2003) 1009–1018.

[42] R. Barandela, F.J. Ferri, J.S. Sanchez, Decision boundary preserving prototype selection for nearest neighbor classification, International Journal of Pattern Recognition and Artificial Intelligence 19 (2005) 787–806.

[43] M.T. Lozano, J.S. Sanchez, F. Pla, Using the geometrical distribution of prototypes for training set condensing, in: CAEPIA, Lecture Notes in Computer Science, vol. 3040, 2003, pp. 618–627.

[44] L.I. Kuncheva, L.C. Jain, Nearest neighbor classifier: simultaneous editing and feature selection, Pattern Recognition Letters 20 (1999) 1149–1156.

[45] L.I. Kuncheva, Editing for the k-nearest neighbors rule by a genetic algorithm, Pattern Recognition Letters 16 (8) (1995) 809–814.

[46] D.B. Skalak, Prototype and feature selection by sampling and random mutation hill climbing algorithms, in: Machine Learning: Proceedings of the Eleventh International Conference, 1994, pp. 293–301.

[47] S. García, J.R. Cano, F. Herrera, A memetic algorithm for evolutionary prototype selection: a scaling up approach, Pattern Recognition 41 (2008) 2693–2709.

[48] J.R. Cano, F. Herrera, M. Lozano, Using evolutionary algorithms as instance selection for data reduction in KDD: an experimental study, IEEE Transactions on Evolutionary Computation 7 (6) (2003) 561–575.

[49] E. Marchiori, Hit miss networks with applications to instance selection, Journal of Machine Learning Research 9 (2008) 997–1017.

[50] D.R. Wilson, T.R. Martinez, Reduction techniques for instance-based learning algorithms, Machine Learning 38 (2000) 257–286.

[51] E. Marchiori, Class conditional nearest neighbor for large margin instance selection, IEEE Transactions on Pattern Analysis and Machine Intelligence 32 (2010) 364–370.

[52] K.-P. Zhao, S.-G. Zhou, J.-H. Guan, A.-Y. Zhou, C-pruner: an improved instance pruning algorithm, in: International Conference on Machine Learning and Cybernetics, vol. 1, 2003, pp. 94–99.

[53] D.W. Aha, D. Kibler, Instance-based learning algorithms, in: Machine Learning, 1991, pp. 37–66.

[54] H. Brighton, C. Mellish, Advances in instance selection for instance-based learning algorithms, Data Mining and Knowledge Discovery 6 (2) (2002) 153–172.

[55] L.A. Zadeh, Fuzzy sets, Information and Control 8 (3) (1965) 338–353.

[56] M. Yang, S. Chen, X. Yang, A novel approach of rough set-based attribute reduction using fuzzy discernibility matrix, in: Proceedings of the Fourth International Conference on Fuzzy Systems and Knowledge Discovery, 2007, pp. 96—101.

[57] B. Sun, Z. Gong, D. Chen, Fuzzy rough set theory for the interval-valued fuzzy information systems, Information Sciences 178 (13) (2008) 2794–2815.
[58] X. Wang, X. Tsang, D. Zhao, D. Chen, D. Yean, Learning fuzzy rules from fuzzy samples based on rough set technique, Information Sciences 177 (20) (2007) 4493–4514.
[59] N. Verbiest, C. Cornelis, R. Jensen, Fuzzy rough positive region-based nearest neighbour classification, in: Proceedings of the 20th International Conference on Fuzzy Systems, 2012, pp. 1961–1967.
[60] J.M. Keller, M.R. Gray, J.R. Givens, A fuzzy k-nearest neighbor algorithm, IEEE Transactions on Systems, Man, and Cybernetics 15 (1985) 580–585.
[61] A. Radzikowska, E. Kerre, A comparative study of fuzzy rough sets, Fuzzy Sets and Systems 126 (2002) 137–156.
[62] J.A. Fernandez, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera, Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework, Journal of Multiple-Valued Logic and Soft Computing 17 (2–3) (2011) 255–287.
[63] J. Cohen, A coefficient of agreement for nominal scales, Educational and Psychological Measurement 20 (1960) 37–46.
[64] J. Demsar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine Learning Research 7 (2006) 1–30.
[65] F. Wilcoxon, Individual comparisons by ranking methods, Biometrics Bulletin 6 (1945) 80–83.
[66] J. Derrac, S. García, D. Molina, F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, Swarm and Evolutionary Computation 1 (1) (2011) 3–18.
[67] S. García, J.A. Fernandez, J. Luengo, F. Herrera, Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power, Information Sciences 180 (10) (2010) 2044–2064.
[68] S. García, A. Fernández, J. Luengo, F. Herrera, A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability, Soft Computing 13 (2009) 959–977.

Nele Verbiest holds an M.Sc. degree in Mathematical Informatics from Ghent University (2010). Currently, she is working on a project funded by the Special Research Fund (BOF) at Ghent University. Her research is on solving data mining problems with fuzzy rough sets.

Chris Cornelis holds an M.Sc. (2000) and a Ph.D. degree (2004) in Computer Science both from Ghent University, Belgium. Currently, he is a postdoctoral fellow at the University of Granada supported by the Ramón y Cajal programme, as well as a guest professor at Ghent University. He has co-authored more than 30 papers published in international journals. He serves as a member of the editorial board of the International Journal of Computational Intelligence Research, and of Transactions on Rough Sets. His current research interests include fuzzy sets and rough sets, instance selection, feature selection, classification and recommender systems.

Francisco Herrera received his M.Sc. in Mathematics in 1988 and Ph.D. in Mathematics in 1991, both from the University of Granada, Spain.

He is currently a Professor in the Department of Computer Science and Artificial Intelligence at the University of Granada. He has had more than 200 papers published in international journals. He is coauthor of the book “Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases” (World Scientific, 2001).

He currently acts as Editor in Chief of the international journal “Progress in Artificial Intelligence” (Springer) and serves as area editor of the Journal Soft Computing (area of evolutionary and bioinspired algorithms) and International Journal of Computational Intelligence Systems (area of information systems). He acts as an Associate Editor of the journals: IEEE Transactions on Fuzzy Systems, Information Sciences, Advances in Fuzzy Systems, and International Journal of Applied Metaheuristics Computing; and he serves as a member of several journal editorial boards, among others: Fuzzy Sets and Systems, Applied Intelligence, Knowledge and Information Systems, Information Fusion, Evolutionary Intelligence, International Journal of Hybrid Intelligent Systems, Memetic Computation, Swarm and Evolutionary Computation.

He received the following honors and awards: ECCAI Fellow 2009, 2010 Spanish National Award on Computer Science ARITMEL to the “Spanish Engineer on Computer Science”, and International Cajastur “Mamdani” Prize for Soft Computing (Fourth Edition, 2010).

His current research interests include computing with words and decision making, data mining, bibliometrics, data preparation, instance selection, fuzzy rule based systems, genetic fuzzy systems, knowledge extraction based on evolutionary algorithms, memetic algorithms and genetic algorithms.