Information Sciences 501 (2019) 155–171

Contents lists available at ScienceDirect

Information Sciences

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

Weight selection strategies for ordered weighted average based fuzzy rough sets

Sarah Vluymans[a,b,c], Neil Mac Parthaláin[d], Chris Cornelis[a,*], Yvan Saeys[a,b]

[a] *Department of Applied Mathematics, Computer Science and Statistics, Ghent University, Belgium*
[b] *Data Mining and Modelling for Biomedicine, VIB Center for Inflammation Research, Ghent, Belgium*
[c] *Department of Computer Science and Artificial Intelligence, University of Granada, Spain*
[d] *Department of Computer Science, Aberystwyth University, Wales, UK*

ARTICLE INFO

Article history:
Received 6 September 2017
Revised 17 December 2018
Accepted 29 May 2019
Available online 29 May 2019

Keywords:
Fuzzy rough set theory
Ordered weighted average
Meta-learning

ABSTRACT

Fuzzy rough set theory models both vagueness and indiscernibility in data, which makes it a very useful tool for application to various machine learning tasks. In this paper, we focus on one of its robust generalisations, namely ordered weighted average based fuzzy rough sets. This model uses a weighted approach in the definition of the fuzzy rough operators. Although its efficacy and competitiveness with state-of-the-art machine learning approaches has been well established in several studies, its main drawback is the difficulty in choosing an appropriate weighting scheme. Several options exist and an adequate choice can greatly enhance the suitability of the ordered weighted average based fuzzy rough operators. In this work, we develop a clear strategy for the weighting scheme selection based upon the underlying characteristics of the data. The advantages of the approach are presented in a detailed experimental study focusing. Rather than to propose a classifier, our aim is to present a strategy to select a suitable weighting scheme for ordered weighted average based fuzzy rough sets in general. Our weighting scheme selection process allows users to take full advantage of the versatility offered by this model and performance improvements over the traditional fuzzy rough set approaches.

1. Introduction

Uncertainty is a pervasive problem in real-world data. Any machine learning method applied to such data must have a mechanism to handle it effectively. One particular way to do so is to use a mathematical tool that models uncertainty. Fuzzy rough set theory [12] offers such advantages. It was proposed as a hybrid form of fuzzy set theory [47] and rough set theory [26] and its central idea is the approximation of fuzzy concepts by means of two fuzzy sets: the fuzzy rough lower and upper approximations. The main limitation of the original fuzzy rough set model is its high sensitivity to noise. In order to address this shortcoming, several noise-tolerant fuzzy rough set models have been proposed in the literature (e.g. review [18]). Both the traditional model and its later extensions have been used successfully in many different areas of machine learning [37], such as feature selection, instance selection, classification and clustering.

[*] Corresponding author at: Department of Applied Mathematics, Computer Science and Statistics, Ghent University, Belgium.
E-mail addresses: sarah.vluymans@ugent.be (S. Vluymans), chris.cornelis@ugent.be (C. Cornelis).

https://doi.org/10.1016/j.ins.2019.05.085
0020-0255/© 2019 Elsevier Inc. All rights reserved.

Context. In this paper, we focus on one of these robust extensions: ordered weighted average (OWA) based fuzzy rough sets [10]. It addresses the noise sensitivity problem of the traditional model by replacing the strict minimum and maximum in its fuzzy rough approximation definitions by appropriate OWA aggregations [44], which represent softened versions of these operators. We focus on this model because: (i) it was identified as the most interesting among the alternatives compared in [11] and (ii) it has been employed with success for several machine learning tasks, such as imbalanced classification [28,35,41], instance selection [33] and missing value imputation [1]. These methods rely on the OWA-based fuzzy rough approximation operators to make class predictions or derive instance or feature quality measures. Although the effectiveness of OWA-based fuzzy rough sets for such techniques has been clearly demonstrated and methods using this model have been shown to outperform the state-of-the-art in these domains, it is not as widely adopted as it could be. We suspect that one of the reasons for this is the requirement to specify the OWA weighting scheme, the crucial component of this model defining the weights used in its fuzzy rough approximation operators. Several options for these weights are encountered in the literature. However, to date, there are no documented guidelines on which weighting scheme should be chosen within OWA-based fuzzy rough sets. The need for a data-driven weighting scheme selection has been pointed out in e.g. [35] and the optimal choice indeed depends on the data at hand.

Aim. In this work, we remedy exactly this situation and provide a transparent strategy for the weighting scheme selection process. Our research belongs to the domain of meta-learning [24] and we study several existing weight definitions and offer explanations as to why some definitions are preferred over others in specific situations. This paper resolves the missing link in the literature between the theoretical foundations and practical applications of OWA-based fuzzy rough set theory. In this way, we make OWA-based fuzzy rough set theory more accessible to other researchers by removing the often cumbersome weighting scheme selection step. Since it is based on simple dataset properties, there is no true increase in computational cost by following our proposed strategy.

Methodology and contributions. In the experimental evaluation, we compare five different weighting schemes. These include four data-independent versions used in previous studies and a new data-dependent setting proposed in this paper. By including a data-dependent version, we can assess whether considering the underlying data distribution can benefit the OWA-based fuzzy rough model. We will show that this setting is useful overall and also the preferred choice for some challenging datasets, such as those containing only nominal-valued features. Our evaluation demonstrates that no weighting scheme stands out in general, such that no single approach can be selected as a default setting. This reinforces the importance of our proposed selection strategy, that specifies which scheme is preferred in which particular situation. We provide intuitive but thorough explanations for the behaviour of the five schemes. This offers further insight into the mechanisms which underpin OWA-based fuzzy rough sets, a second fundamental contribution of this work. Some conclusions drawn in this paper are expected to transfer to other applications of OWA operators (including multi-criteria and multi-person decision making [46]) as well, where the weight selection process is always a critical step.

Structure. The remainder of this paper is organized as follows. In Section 2, we recall the definitions of both traditional and OWA-based fuzzy rough sets. Section 3 defines the weighting schemes and offers a comparison between them. In Section 4, we present our proposed strategy for the OWA weight selection process and discuss the benefits and limitations of the different schemes. An important point is the validation of our proposal. This is carried out in Section 5 and includes the essential evaluation on a series of independent datasets and in different machine learning applications. Finally, Section 6 draws some conclusions.

2. OWA-based fuzzy rough sets

In this section, we recall the motivation and definition of OWA-based fuzzy rough sets. Section 2.1 describes the traditional fuzzy rough set model and Section 2.2 details the OWA-based generalization. Section 2.3 summarizes how the OWA-based model has been used in various machine learning methods.

2.1. Fuzzy rough set theory

Fuzzy rough set theory was proposed in [12] as a hybrid model of fuzzy set theory [47] and rough set theory [26]. Both deal with uncertainty in data, but from different perspectives. Fuzzy set theory models vague or ill-defined concepts (e.g. the set of young people) by allowing partial memberships of objects to a set. Such a partial membership degree is a real number between 0 and 1, where the two extremes are interpreted as either complete exclusion from or inclusion in the set. Rough sets manage data indiscernibility, the situation where the observed features are insufficient to sharply delineate a concept. Rough set theory approximates such an incomplete concept A in two ways. The lower approximation groups all objects certainly belonging to A, while the upper approximation consists of objects possibly belonging to A.

Fuzzy rough set theory introduces fuzziness into rough sets and allows the approximated concept as well as the lower and upper approximations to be modelled as fuzzy rather than crisp sets. To measure the similarity between two objects, a fuzzy relation $R(\cdot, \cdot)$ is used. We consider the implicator/t-norm fuzzy rough set model [27]. An implicator $\mathcal{I}: [0,1]^2 \to [0,1]$ is a fuzzy operator that is decreasing in its first argument, increasing in the second and satisfies the boundary conditions $\mathcal{I}(0,0)=\mathcal{I}(0,1)=\mathcal{I}(1,1)=1$ and $\mathcal{I}(1,0)=0$. A triangular norm (t-norm) $\mathcal{T}:[0,1]^2\to[0,1]$ is a commutative and associative fuzzy operator that is increasing in both arguments and satisfies the boundary condition $(\forall a\in[0,1])(\mathcal{T}(a,1)=a)$. Let A be the (fuzzy) set to approximate. The membership degree of x to the lower approximation of A is defined as

$$\underline{A}(x)=\min_{y\in X}[\mathcal{I}(R(x,y),A(y))], \tag{1}$$

where X is the full dataset. The membership degree to the upper approximation is given by

$$\overline{A}(x)=\max_{y\in X}[\mathcal{T}(R(x,y),A(y))]. \tag{2}$$

In this paper (and others using fuzzy rough set theory in machine learning), the set A corresponds to a decision class and is therefore non-fuzzy. The values $A(\cdot)$ can only be 1 or 0, depending on whether or not an element belongs to A. Taking this into account, expression (1) can be rewritten as

$$\begin{aligned}\underline{A}(x)&=\min_{y\in X}[\mathcal{I}(R(x,y),A(y))]=\min\left[\min_{y\in A}[\mathcal{I}(R(x,y),1)],\min_{y\notin A}[\mathcal{I}(R(x,y),0)]\right]\\&=\min\left[1,\min_{y\notin A}[\mathcal{I}(R(x,y),0)]\right]=\min_{y\notin A}[\mathcal{I}(R(x,y),0)]=\min_{y\notin A}[\mathcal{N}_{\mathcal{I}}(R(x,y))],\end{aligned} \tag{3}$$

where $\mathcal{N}_{\mathcal{I}}:[0,1]\to[0,1]$ is the induced negator of $\mathcal{I}$, defined as $(\forall a)(\mathcal{N}_{\mathcal{I}}(a)=\mathcal{I}(a,0))$. In this derivation, we have used the fact that $(\forall a)(\mathcal{I}(a,1)=1)$, which is due to the boundary condition $\mathcal{I}(1,1)=1$ and the implicator being decreasing in its first argument. In a similar way, expression (2) reduces to

$$\overline{A}(x)=\max_{y\in A}[R(x,y)]. \tag{4}$$

The fuzzy relation used in this paper is defined as follows. Let x and y be two elements in the dataset and $\mathcal{F}$ the feature set. The similarity between x and y is computed as

$$R(x,y)=\frac{1}{|\mathcal{F}|}\sum_{f\in\mathcal{F}}R_f(x,y). \tag{5}$$

The relation $R_f(\cdot,\ \cdot)$ measures the similarity between elements based on feature f. For a numeric feature, this relation is defined as $R_f(x,y)=1-\frac{|x_f-y_f|}{range(f)}$, where x_f and y_f are the values of x and y for feature f and the denominator $range(f)$ is its range. When f is a nominal feature, we set $R_f(x,\ y)$ to 1 when $x_f=y_f$ and to 0 otherwise. We have selected relation (5) because it has shown a good behaviour in related studies (e.g. [11,28,32]). Some alternatives can be found in e.g. [20]. As we will argue in Section 4.5, we believe that the conclusions drawn in this paper carry over to those settings as well.

2.2. OWA-based model

As evident from (3) and (4), the fuzzy rough approximations $\underline{A}(x)$ and $\overline{A}(x)$ depend on the similarity of x with a single element y. This makes the traditional fuzzy rough set model highly susceptible to noise. Several noise-tolerant fuzzy rough set models have been proposed in the literature, including fuzzy variable precision rough sets [48], vaguely quantified fuzzy rough sets [9], β-precision fuzzy rough sets [18] and a data distribution aware model [3]. In this paper, we turn our attention to OWA-based fuzzy rough sets [10], which have been shown to be a preferred alternative among noise-tolerant models [11,39]. The noise sensitivity of (3) and (4) is addressed by replacing the min and max operators in these definitions by ordered weighted average (OWA, [44]) aggregations.

Definition 1. Given a set of values $V=\{a_1,a_2,\ldots,a_p\}$ and a weight vector $W=\langle w_1,w_2,\ldots,w_p\rangle$ with $(\forall w_i)(w_i\in[0,1])$ and $\sum_{i=1}^p w_i=1$. The OWA aggregation of V using weight vector W is defined as $\mathrm{OWA}_W(V)=\sum_{i=1}^p(w_ib_i)$, where b_i is the ith largest value in V.

As Definition 1 indicates, the first step in an OWA process is to sort the values which are to be aggregated. Afterwards, the weights in W are assigned to the values in the ordered sequence and a weighted average is computed. In the OWA-based fuzzy rough set model, the minimum and maximum in (3) and (4) are replaced by appropriate OWA aggregations with weight vectors W_L and W_U respectively, such that $orness(W_U)\geq\frac{1}{2}\geq orness(W_L)$. The orness measure is a value between 0 and 1 and expresses how similar the weight vector is to the strict maximum. The membership degree to the lower approximation is computed as $\underline{A}(x)=\mathrm{OWA}_{W_L}(\{\mathcal{N}_{\mathcal{I}}(R(x,y)\,|\,y\notin A\})$. We use the standard negator $(\mathcal{N}_{\mathcal{I}}(x)=1-x)$, which is the induced negator of such popular implicators as the Kleene-Dienes, Łukasiewicz and Reichenbach implicators [22]. Consequently, we derive

$$\underline{A}(x)=\mathrm{OWA}_{W_L}(\{1-R(x,y)\,|\,y\notin A\}). \tag{6}$$

For the upper approximation, we find

$$\overline{A}(x)=\mathrm{OWA}_{W_U}(\{R(x,y)\,|\,y\in A\}). \tag{7}$$

By actively including more than one element y in the calculation of $\underline{A}(x)$ and $\overline{A}(x)$, the sensitivity to noise and outliers is reduced. This results in the superior noise-tolerance of the OWA-based fuzzy rough set model compared to the traditional one. Most commonly, W_L consists of increasing weights, such that the largest weight is associated with the minimum, thereby softening the min operator. For the same reason, W_U usually contains decreasing weights. A so-called *weighting scheme* provides the definitions of W_L and W_U. We list and discuss several alternatives in Section 3.

2.3. Applications

The survey paper [37] discusses applications of fuzzy rough set theory in various machine learning domains. As noted in the introduction, the OWA-based model has been used in, among others, classification, instance selection and missing value imputation. These methods internally rely on the OWA-based approximations (6) and (7).

- **Classification:** in [28] and [41], classifiers for imbalanced data were developed, focusing on two-class imbalanced datasets. They compute the membership degree of a target instance to the OWA-based lower approximation of both classes and assign it to the class for which the derived value is largest. The proposed methods deal with the class imbalance problem by using class-dependent weighting schemes. The method of [28] was extended to multi-class imbalanced classification in [38]. Recently, the contribution of [36] proposed a multi-label classifier, applying an OWA-based fuzzy rough neighbourhood consensus.
- **Instance quality:** the proposals of [2], [33] and [35] compute the quality of an instance based on its membership degree to the OWA-based lower and upper approximations of its own class. In [35], the instance quality values are used within a nearest neighbour classifier and assign a more important vote to high quality instances in the class prediction step. The studies of [2] and [33] develop instance selection methods using wrapper approaches.
- **Missing value imputation:** the authors of [1] proposed missing value imputation methods based on the combination of three fuzzy rough set models, including the OWA-based alternative, with a nearest neighbour classifier. The lower and upper approximations of classes are used internally in the nearest neighbour method, as done in [19].

3. OWA weighting schemes

Expressions (6) and (7) show that the OWA-based fuzzy rough lower and upper approximations depend on weight vectors W_L and W_U respectively and the weighting scheme that defines them. We list some existing weight definitions in Section 3.1 and introduce a new data-dependent version in Section 3.2. In Section 3.3, we offer a first comparison between the characteristics of the different schemes.

3.1. Data-independent schemes

Most weight definitions used in applications of OWA-based fuzzy rough sets only depend on p, the length of the weight vector. We list four of them in this section. For the lower approximation of A as defined in (6), p equals the size of the complement of A. For the upper approximation (7), p equals the size of A. In this section, for a fixed value of p, the vectors W_L and W_U are reversals of each other. Keeping this in mind, we only specify the definition of W_L below.

Strict weights (Strict). As a first option, we include a weight setting that is equal to the traditional model from Section 2.1, for which the lower approximation weight vector is $W_L^{strict} = \langle 0, 0, \ldots, 0, 1\rangle$. By only assigning a non-zero weight to the last position, the exact minimum is obtained in (6). It should be clear that the traditional model (3) is indeed a special case of the OWA-based model.

Additive weights (Add). The weight vector for the lower approximation is given by

$$W_L^{add} = \left\langle \frac{2}{p(p+1)}, \frac{4}{p(p+1)}, \ldots, \frac{2(p-1)}{p(p+1)}, \frac{2}{p+1} \right\rangle. \tag{8}$$

These weights are the normalized version of the vector $\langle 1, 2, \ldots, p-1, p\rangle$. The normalization was carried out to satisfy the conditions in Definition 1. Each weight w_{i+1} is obtained by adding the constant value $\frac{2}{p(p+1)}$ to the previous weight w_i. This means that every next value is assumed to have a constant increase in its relevance to determine the aggregated value.

Exponential weights (Exp). The weight vector for the lower approximation is given by

$$W_L^{exp} = \left\langle \frac{1}{2^p-1}, \frac{2}{2^p-1}, \ldots, \frac{2^{p-2}}{2^p-1}, \frac{2^{p-1}}{2^p-1} \right\rangle. \tag{9}$$

The weight w_{i+1} is determined by multiplying w_i by the constant factor 2. Every value is deemed twice as relevant for the aggregation as the previous one.

Inverse additive weights (Invadd). The weight vector for the lower approximation is given by

$$W_L^{invadd} = \left\langle \frac{1}{pD_p}, \frac{1}{(p-1)D_p}, \ldots, \frac{1}{2D_p}, \frac{1}{D_p} \right\rangle, \tag{10}$$

with $D_p = \sum_{i=1}^{p} \frac{1}{i}$, the pth harmonic number. This vector is the normalized version of $\langle \frac{1}{p}, \frac{1}{p-1}, \ldots, \frac{1}{2}, 1 \rangle$. For this setting, as opposed to the other two discussed above, the relation between w_i and w_{i+1} depends on i. To obtain w_{i+1}, w_i is multiplied with the factor $\frac{p-i+1}{p-i}$, which increases with i.

3.2. Data-dependent schemes

The four settings detailed in Section 3.1 only depend on the aggregation length and do not take the values to be aggregated into consideration. Although equal-sized sets can contain very different values, the weights used to aggregate them will be the same. In this section, we propose an alternative setting, called *Mult*, which bases its weights on the values to be aggregated.

Let $V = \langle v_1, \ldots, v_n \rangle$ be the sorted set of values to aggregate. This implies that v_n is the smallest value among them. For any other value, its similarity with v_n can be computed as $s(v_i) = 1 - |v_i - v_n|$. Since all values in V belong to the unit interval, all values $s(v_i)$ do so as well. Based on these similarity values, we can define a function (for v_1 to v_{n-1})

$$m(v_i) = \begin{cases} 1 & \text{if } v_i = v_{i+1}, \\ s(v_i) & \text{if } v_i \neq v_{i+1}. \end{cases} \tag{11}$$

Mult constructs its weights from w_n to w_1, that is, from the largest to the smallest weight. As a first step, w_n is set to 1. Next, w_i is calculated by multiplying w_{i+1} by the factor $m(v_i)$. The vector obtained by this procedure is

$$W_L^{mult*} = \left\langle \prod_{i=1}^{n-1} m(v_i), \prod_{i=2}^{n-1} m(v_i), \ldots, m(v_{n-2}) \cdot m(v_{n-1}), m(v_{n-1}), 1 \right\rangle. \tag{12}$$

In order to satisfy the conditions in Definition 1, the final vector W_L^{mult} is the normalized version of W_L^{mult*}.

As in *Invadd*, consecutive weights differ by a factor that depends on i. To determine w_i, weight w_{i+1} is multiplied by $m(v_i)$. By using $m(v_i)$ rather than $s(v_i)$, we ensure that when the values v_i and v_{i+1} are the same, they are assigned the same weight. If the two values differ, w_{i+1} is multiplied by the factor $s(v_i)$. This factor decreases when i decreases, because values earlier in the ordered sequence V are less similar to the minimum value v_n. The decreasing factor means that the weights drop more rapidly toward the beginning of the vector. This is a desirable property, as we expect that the first values in V are far less relevant to the aggregation (a softened minimum) than the later ones.

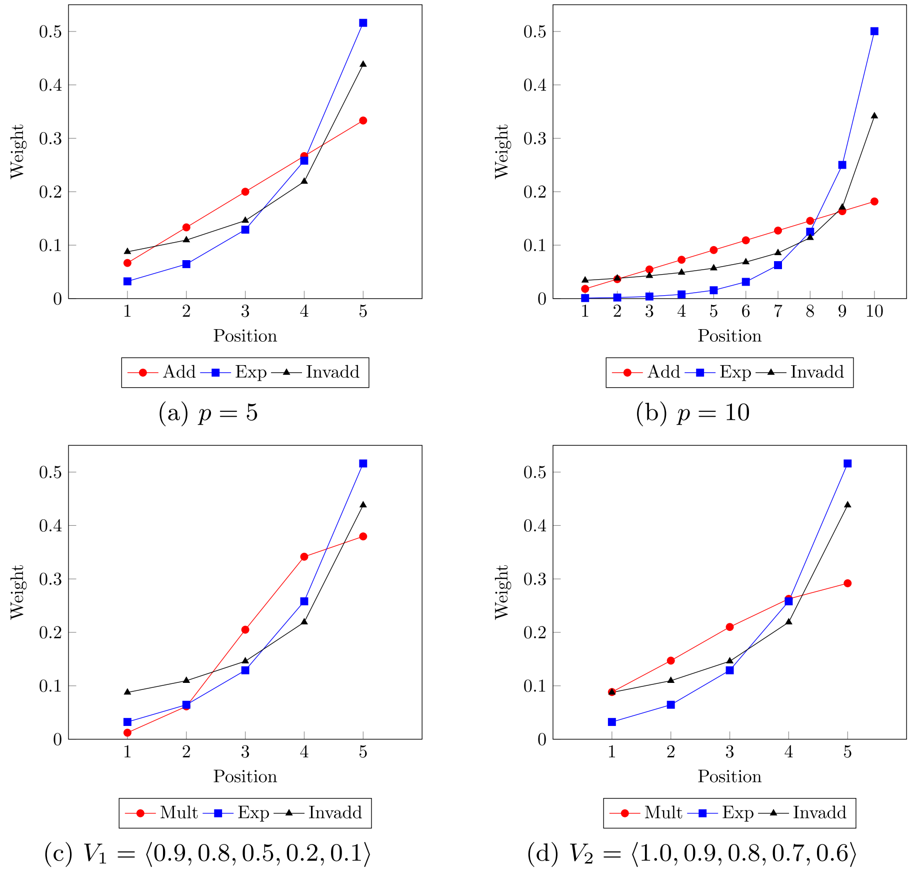
Mult is the only data-dependent weight setting included in our study. We can find some advances on learning OWA weights from data in the literature, although not in the context of OWA-based fuzzy rough sets. In [25], an OWA weight generation procedure is proposed that maximizes the dispersion of the weights for a given orness value. An analytic solution is offered in [15]. An orness value needs to be specified. Alternatively, the weights can be calculated when a number of samples, in the form of p values and their associated aggregation outcome, are provided. Examples of this approach can be found in [4,5,14,31,45]. Although interesting, these methods are of no use to us here, because we cannot train the weights with a given orness value or known aggregation outcomes, simply because it is not clear how these parameters should be set.

A dependent OWA weight vector was proposed in [43]. This method models a weighted mean of the aggregation values instead of a minimum. The weight of a value is related to its distance to the mean of all values. In [6], a cluster-based OWA aggregation was proposed. To aggregate a set of values, the reliability of each value to the entire set is evaluated based on a clustering of all values. A shortcoming of this procedure is its high computational cost as a result of the clustering step. This was pointed out by [7], which proposed a more efficient procedure to determine the reliability of a value. Like *Mult*, these methods compute their weights based on the values to be aggregated. Nevertheless, we cannot use them within the OWA-based fuzzy rough approximations, because they do not act as a softened minimum, but model averages instead.

3.3. Weighting scheme comparison

We study five weighting schemes in this paper: the traditional model *Strict*, the data-independent versions *Add, Exp* and *Invadd* and our data-dependent proposal *Mult*. The computational complexity to aggregate p values with an OWA procedure is $\mathcal{O}(p\log(p))$ due to the cost of the sorting step. Sorting the values is not required in *Strict*, such that its cost reduces to $\mathcal{O}(p)$.

In Fig. 1, we illustrate the weights W_L generated by the included alternatives for some sets with a small number of values to be aggregated. On the horizontal axis, we plot the position of a value in the ordered sequence. The vertical axis represents the weight at a particular position. It is clear that these weight vectors are used to soften the minimum, as they put more emphasis on the higher positions in the ordered sequence, which correspond to lower values (Definition 1). We



(a) $p = 5$

(b) $p = 10$

(c) $V_1 = \langle 0.9, 0.8, 0.5, 0.2, 0.1 \rangle$

(d) $V_2 = \langle 1.0, 0.9, 0.8, 0.7, 0.6 \rangle$

Fig. 1. Illustration of the different weight settings.

remind the reader that *Strict* always assigns a weight 1 to the last position and weight 0 to all others. This setting is not plotted.

In Fig. 1a and b, we compare the three data-independent settings (*Add, Exp* and *Invadd*) for sets with size 5 and 10 respectively. As was clear from their description, the additive weights take on the form of a straight line with slope $\frac{2}{p(p+1)}$, while the exponential weights are taken from an exponential function with base 2. The relative weight increase for *Exp* is the same in each step, that is, w_{i+1} is obtained by multiplying w_i with a constant factor 2. For the inverse additive weights, this relative increase becomes larger when the position i increases. It is clear from Fig. 1a and b that this results in higher weights for the lower positions for *Invadd* compared to *Exp*. The exponential weights cancel out the contribution of the values at the lowest positions (in particular when p is high), while *Invadd* always assigns them a non-negligible weight. This is an important difference between these two settings, which will be discussed further in Section 4. *Add* divides the weights more evenly across the positions than any other setting. This may not always be beneficial. Especially when p is large, the weight associated with the true minimum may be relatively too low. For large values of p, *Add* becomes closely related to a regular average.

We note that, due to the choice of negator in (6), the *Strict* and *Exp* settings are related to a nearest neighbour approach. For *Strict*, the membership degree of x to the lower approximation of A is computed as $\underline{A}(x) = \min_{y \notin A}[1 - R(x, y)]$. This procedure locates the nearest neighbour of x that does not belong to A. In *Exp*, the relation with a nearest neighbour technique is more subtle, but is further pronounced as the aggregation length p increases. As stated above, when p increases, *Exp* sets several of the lowest weights to zero. For example, when $p = 50$, only 14 weights in the exponential vector are non-zero. This means that only the 14 nearest elements to x that do not belong to A are used in the calculation of $\underline{A}(x)$. This nearest neighbour characteristic is one of the important aspects which helps to explain the results presented in Section 4.

Fig. 1c and d compare *Mult* to *Exp* and *Invadd*. These alternatives are related to each other in the sense that a weight can be obtained from the previous or next one by multiplication. The factor in this multiplication is fixed in *Exp*, while it varies for *Invadd* and *Mult*. We consider two different value sets: $V_1 = \langle 0.9, 0.8, 0.5, 0.2, 0.1 \rangle$ and $V_2 = \langle 1.0, 0.9, 0.8, 0.7, 0.6 \rangle$. As discussed above, since V_1 and V_2 have the same size, the exponential and inverse additive do not differ for these two aggregations. The *Mult* setting on the other hand uses the values in V_1 and V_2 to determine its weights. The figures show that these are very different for V_1 and V_2 and closely follow the distribution of their values.

Table 1
Description of the 50 datasets used in the experiments. We list the number of features (nFeat), the number of instances (nInst), the number of classes (nCl) and the IR. Together with the number of features, we specify whether they are all nominal (Y) or not (N).

Name	nFeat	nInst	nCl(IR)	Name	nFeat	nInst	nCl(IR)
abalone	8(N)	4174	28(689)	page-blocks	10(N)	5472	5(175.46)
australian	14(N)	690	2(1.25)	phoneme	5(N)	5404	2(2.41)
automobile	25(N)	159	6(16)	pima	8(N)	768	2(1.87)
balance	2(N)	625	3(5.88)	ring	20(N)	7400	2(1.02)
banana	2(N)	5300	2(1.23)	saheart	9(N)	462	2(1.89)
bands	19(N)	365	2(1.7)	satimage	36(N)	6435	6(2.45)
breast	9(Y)	277	2(2.42)	segment	19(N)	2310	7(1)
bupa	6(N)	345	2(1.38)	sonar	60(N)	208	2(1.14)
car	6(Y)	1728	4(18.62)	spambase	57(N)	4597	2(1.54)
cleveland	13(N)	297	5(12.62)	spectfheart	44(N)	267	2(38.56)
contra	9(N)	1473	3(1.89)	splice	60(Y)	3190	3(2.16)
crx	15(N)	653	2(1.21)	texture	40(N)	5500	11(1)
derma	34(N)	358	6(5.55)	thyroid	21(N)	7200	3(40.16)
ecoli	7(N)	336	8(28.6)	tic-tac-toe	9(Y)	958	2(1.89)
flare	11(Y)	1066	6(7.7)	titanic	3(N)	2201	2(2.1)
german	20(N)	1000	2(2.34)	twonorm	20(N)	7400	2(1)
glass	9(N)	214	6(8.44)	vehicle	18(N)	846	4(1.1)
haberman	3(N)	306	2(2.78)	vowel	13(N)	990	11(1)
heart	13(N)	270	2(1.25)	wdbc	30(N)	569	2(1.68)
ionosphere	33(N)	351	2(1.79)	wine	13(N)	178	3(1.48)
mammo	5(N)	830	2(1.06)	winequal-r	11(N)	1599	6(68.1)
marketing	13(N)	6876	9(2.49)	winequal-w	11(N)	4898	7(439.6)
monk-2	6(N)	432	2(1.12)	wisconsin	9(N)	683	2(1.86)
mov_lib	90(N)	360	15(1)	yeast	8(N)	1484	10(92.6)
nursery	8(Y)	12,690	5(2160)	zoo	16(Y)	101	7(10.25)

4. Weighting scheme selection strategy

In this section, we analyse the performance of the OWA-based lower approximation predictor using the five weighting schemes described in Sections 3.1 and 3.2. Section 4.1 lays out the details of our experimental study and Section 4.2 presents an initial high-level comparison of the different OWA weighting schemes. In Section 4.3, we divide the included datasets into eight groups, based on which we can present a selection strategy for the OWA weighting scheme. We explain the observed behaviour of the different weight settings in Section 4.4. Section 4.5 groups some remarks on our chosen approach.

4.1. Experimental set-up

We compare the different weighting schemes within OWA-based fuzzy rough sets in a classification setting. We follow [18] and use the lower approximation operator as predictor. To classify an instance x, this classifier computes the membership degree $\underline{C}(x)$ of this element for all classes C and assigns x to the class for which this value is highest. It uses expression (6) in this calculation, setting weight vector W_L to one of the five alternatives listed in Sections 3.1 and 3.2. By doing so, we obtain experimental results of the *Strict, Add, Exp, Invadd* and *Mult* weighting schemes showing how well they can separate natural groups (classes) of observations. Note that the aim of this paper is not to propose a new classification method, but rather to develop a strategy to select the weighting scheme for OWA-based fuzzy rough sets within machine learning algorithms in general. Our conclusions transfer to other applications as well (see Section 5.4).

We evaluate the performance on 50 datasets (Table 1) by means of 10-fold cross validation. The datasets and partitions were obtained from the KEEL repository at www.KEEL.es. For each dataset, we list the number of instances, features and classes and specify whether all features are nominal (categorical) or not. Along with the number of classes, we indicate the level of imbalance between them with the imbalance ratio (IR). We compute this measure as the ratio of the sizes of the largest and smallest classes. For two-class datasets, this coincides with the measure traditionally used in studies on class imbalance [29]. The classification performance of the fuzzy rough lower approximation is evaluated by the balanced accuracy. This metric is defined as the mean of the class-wise accuracies and is not negatively affected by class imbalance. Table 1 shows that several datasets are severely imbalanced. It is accepted within the machine learning community that the traditional global accuracy can provide misleading results on such datasets and should therefore be avoided (e.g. [29]).

4.2. Preliminary performance comparison

The results for each dataset can be found in Tables 2 and 3. These tables divide the datasets in several groups, which will be described in Section 4.3. The mean balanced accuracy of the fuzzy rough lower approximation predictor is 0.6693 (*Strict*), 0.6282 (*Add*), 0.6891 (*Exp*), 0.6867 (*Invadd*) and 0.6871 (*Mult*). The strict model is the best setting for 11 datasets,

Table 2
Balanced accuracy results for the datasets in the first five groups of datasets.

Dataset	Strict	Add	Exp	Invadd	Mult
Only nominal features					
breast	0.4946	**0.5908**	0.5537	0.5826	0.5783
car	0.2459	0.3732	0.2638	0.3474	**0.3892**
flare	0.1971	0.4179	0.3107	**0.4731**	0.4274
nursery	0.2267	**0.2276**	0.2004	0.2268	0.2240
splice	0.5491	**0.5972**	0.5416	0.5737	0.5770
tic-tac-toe	0.5049	**0.5902**	0.5211	0.5658	0.5659
zoo	0.9229	0.8526	0.8883	0.8883	**0.9621**
Mean	0.4487	0.5214	0.4685	0.5225	**0.5320**
Perfectly balanced, low complexity (high Fisher score)					
mov_lib	**0.8722**	0.5889	0.8678	0.8500	0.8589
segment	**0.9766**	0.8433	0.9745	0.9494	0.9649
texture	0.9869	0.7596	**0.9884**	0.9664	0.9775
vowel	**0.9939**	0.6000	0.9859	0.9788	0.9869
Mean	**0.9574**	0.6980	0.9541	0.9361	0.9470
At least 30 features, at most 1000 instances					
derma	0.9554	0.8844	**0.9765**	0.9689	0.9722
ionosphere	**0.8777**	0.7321	0.8684	0.8059	0.8311
sonar	0.8515	0.7962	0.8592	**0.8928**	0.8569
spectfheart	0.6165	0.6387	0.6295	**0.7310**	0.7100
wdbc	0.9507	0.9313	**0.9655**	0.9412	0.9473
Mean	0.8503	0.7965	0.8598	**0.8679**	0.8635
More than five classes, IR ≤ 10					
glass	0.7106	0.4659	**0.7130**	0.6039	0.6236
marketing	0.2101	0.2307	0.2575	**0.2701**	0.2653
satimage	0.8946	0.6360	**0.9026**	0.8309	0.8707
Mean	0.6051	0.4442	**0.6244**	0.5683	0.5865
More than five classes, IR > 10					
abalone	0.1116	0.0903	0.1150	0.1047	**0.1364**
automobile	**0.7471**	0.5243	0.6734	0.6550	0.6600
ecoli	0.7170	0.5598	0.7413	0.7381	**0.7476**
winequal-r	**0.3616**	0.2811	0.3590	0.3355	0.3561
winequal-w	**0.4498**	0.2567	0.4419	0.3863	0.4067
yeast	0.5181	0.3960	0.5359	0.5554	**0.5564**
Mean	**0.4842**	0.3514	0.4777	0.4625	0.4772

Add for 8, *Exp* and *Mult* for 9 and *Invadd* for 14. When we derive our weight selection strategy, we do not wish to overfit these results by only focusing on the best setting for each dataset. Instead, we interpret any result that is within 0.05 of the best value as acceptable and any other as poor. We can observe that *Strict* performs poorly on 18 datasets, *Add* on 15, *Exp* on 12 and *Invadd* and *Mult* on 10.

Based on their average performance, we would conclude that (i) *Exp, Invadd* and *Mult* are competitive weight settings and outperform *Strict* and (ii) *Add* does not work well. However, we observe that *Exp, Invadd* and *Mult* perform poorly on at least one fifth of the datasets, meaning that it is not a good idea to select one of these alternatives as a default option. It is also not prudent to exclude *Add* from consideration, as it gives the best result on eight datasets. Clearly, the optimal weighting scheme differs between datasets. The study of this phenomenon is our focus. Below, we discuss why certain weighting schemes are preferred in specific situations and present a clear selection strategy.

4.3. Proposed strategy

In Tables 2 and 3, the 50 datasets are divided into eight groups, based on simple and easy-to-compute data characteristics. For each dataset, the highest value is printed in bold typeface and any value that is more than 0.05 lower is underlined. Values that are not underlined are considered to correspond to an acceptable alternative to the best setting. For each group of datasets, one particular weighting scheme emerges as the best performing one. These are framed in Tables 2 and 3. We do not only focus on the best performing settings, because we do not wish to overfit this data by proposing strategies that are too specific. We encounter the following groups of datasets, along with their corresponding recommended weighting schemes:

1. **Datasets with only nominal features:** in our similarity relation (5), the feature-wise similarity $R_f(x, y)$ between elements x and y based on a nominal feature f can only be 0 or 1, depending on whether these elements take on the same value for f. As a result of this lack of variety in similarity values, we can expect that many elements are found at the same distance

Table 3
Balanced accuracy results for the datasets in the last three groups of datasets.

Dataset	Strict	Add	Exp	Invadd	Mult
At most five classes, at most 4000 instances, IR ≤ 2					
australian	0.7250	**0.8730**	0.7946	0.8675	0.8513
bands	**0.7308**	0.6509	0.7164	0.7173	0.6439
bupa	0.6160	0.6422	0.6498	**0.6782**	0.6696
contra	0.4030	0.4690	0.4374	**0.4836**	0.4604
crx	0.7715	0.8695	0.8326	**0.8706**	0.8650
heart	0.7808	0.8225	0.8058	**0.8242**	0.8117
mammo	0.7456	0.8205	0.8082	**0.8214**	0.8114
monk-2	0.7409	0.9052	0.9059	**0.9171**	0.8144
pima	0.6516	0.7014	0.6754	**0.7040**	0.6778
saheart	0.5853	0.6620	0.6104	**0.6769**	0.6315
vehicle	0.6942	0.5670	**0.7082**	0.6615	0.7006
wine	0.9631	0.9631	0.9673	**0.9679**	**0.9679**
wisconsin	0.9617	0.9301	0.9654	0.9497	**0.9737**
Mean	0.7207	0.7597	0.7598	**0.7800**	0.7599
At most five classes, at most 4000 instances, IR > 2					
balance	0.5417	0.7576	0.6378	0.6528	**0.7874**
cleveland	**0.3001**	0.2919	0.2855	0.2820	0.2710
german	0.5619	**0.6576**	0.5750	0.5740	0.5629
haberman	0.5360	**0.6363**	0.5475	0.5651	0.5267
titanic	0.5211	0.6997	0.6812	**0.7109**	0.7083
Mean	0.4922	**0.6086**	0.5454	0.5570	0.5712
At most five classes, more than 4000 instances					
banana	0.8728	0.7246	0.8929	0.8848	**0.8975**
page-blocks	0.7610	0.3198	**0.7915**	0.5506	0.5978
phoneme	0.8738	0.7730	**0.8743**	0.8165	0.8455
ring	**0.7181**	0.5000	0.6924	0.5139	0.5742
spambase	0.8987	0.8027	**0.9091**	0.8730	0.8447
thyroid	**0.6219**	0.5280	0.5928	0.5720	0.4340
twonorm	0.9462	**0.9757**	0.9619	0.9754	0.9723
Mean	0.8132	0.6605	**0.8164**	0.7409	0.7380

of a given element. This renders a nearest neighbour approach unsuitable, as reflected by the poor results obtained by *Strict* and *Exp*.

Our *Mult* proposal is able to model the distinct staircase structure in the values to be aggregated and guarantees that equal values are assigned equal weights. Its data-dependent nature makes *Mult* an appropriate choice for datasets with only nominal features, as evidenced by the results in Table 2. We recommend its use for this group. The results of *Add* and *Invadd* are close together and acceptable on average, but they both perform poorly on at least one dataset. For *Add*, which fails on the *flare* and *zoo* datasets, this is explained by the fact that these datasets have a high number of classes (Section 4.4.1). When a small dataset with only nominal features contains only a few classes, *Add* could be used.

2. **Perfectly balanced datasets with low complexity:** this group contains four datasets, for which all classes have the same size. They also have a low data complexity, which can for instance be evaluated by the multi-class Fisher discriminant score [17]. This metric is defined for datasets with only numeric features. A higher value corresponds to a lower complexity. The four datasets in this group have scores of 2.5413 (*mov_lib*), 15.6143 (*segment*), 10.2872 (*texture*) and 2.0389 (*vowel*), while the average score of the datasets with only numeric features is 1.8451. We recommend the use of *Strict* in this case. On these easy datasets, OWA-based fuzzy rough sets do not have a clear advantage over the traditional model. By selecting *Strict*, the sorting step in the OWA aggregation is avoided. We note that the additive scheme performs very poorly, but this is due to the high number of classes in these datasets (Section 4.4.1). The complexity condition for this group is crucial. For example, dataset *mammo* is almost balanced, but *Strict* gives a poor result. Its Fisher score is 0.4926.
3. **Datasets with at least 30 features and at most 1000 instances:** five datasets are included in this group. The *mov_lib* dataset from group 2 could be included as well, because it contains 360 instances described by 90 features. We also note that these datasets have only numeric features. Due to their high-dimensional nature, elements are pushed close together (empty space phenomenon). This implies that the values aggregated in the OWA step are more similar to each other than expected in lower dimensional datasets. *Add* clearly fails in this situation, since it is most related to an average (Section 3.3). For each class, the values in the aggregation will be very similar. If we combine them with a procedure similar to an average, the final values for all classes will be more or less the same. As a result, the prediction by *Add* is close to a random guess.

   The average results of *Strict, Exp, Invadd* and *Mult* are close together. We recommend the use of *Mult*, because it does not perform poorly on any of these datasets. The three other options sometimes give a bad result. Although *Mult* never obtains the highest balanced accuracy, it is the safest choice. We developed *Mult* in such a way that its weights follow the distribution of the values to be aggregated. We see the benefit of this idea for these complex datasets, where the weights capture important differences and can better discern between classes.

4. **Datasets with more than five classes and IR** ≤ 10**:** we recommend the use of *Exp* for this group (see Sections 4.4.1 and 4.4.3).
5. **Datasets with more than five classes and IR** > 10**:** the traditional model performs best and we recommend the use of *Strict* (see Sections 4.4.1 and 4.4.3).
6. **Datasets with at most five classes, at most 4000 instances and IR** ≤ 2**:** for these small and balanced datasets, the inverse additive scheme stands out as best performing. We observe that there is only one truly bad weighting scheme for this group, namely *Strict*. These datasets are relatively easy to handle, because they are not too large, do not have too many classes and are not too imbalanced. This seems to be a setting in which any OWA aggregation performs better than the strict model, as there are no factors that can severely hinder the OWA procedure.
   The four true OWA aggregations all provide relatively good results. Their average balanced accuracies are not very different. Nevertheless, *Add, Exp* and *Mult* fail on some of these datasets, while *Invadd* never does. Consequently, we advise the use of the latter for this group. We note that *Invadd* has also come out as best general performing setting in previous studies (e.g. [32,40]) and its strength is most evident for this group of datasets. From the 14 datasets in which *Invadd* attained the highest balanced accuracy, nine are contained in this group. When there are no prior challenging factors (e.g. large size, many classes, class imbalance), *Invadd* seems to be a good default choice.
7. **Datasets with at most five classes, at most 4000 instances and IR** > 2**:** *Add* shows the best performance on this group of datasets. Their strength in the presence of class imbalance is explained in Section 4.4.3.
8. **Datasets with at most five classes and more than 4000 instances:** when a dataset contains many instances, using exponential weights is a good option. We explain this behaviour in Section 4.4.2.

We realise that the thresholds selected to create these groups may seem artificial, but they are based on the results in Tables 2 and 3 and will be justified in Section 4.4 and validated in Section 5. The user can also relax these guidelines, for instance by replacing 'more than five classes' by 'many classes', 'at most 4000 instances' by 'a small dataset' and so on. Our main priority is to capture the general behaviour of the weight settings. In order to apply our proposed strategy in practice in a machine learning method using OWA-based fuzzy rough sets, a weighting scheme selection step should be implemented before the main algorithm. Based on the included easy-to-compute dataset characteristics, the method would first evaluate whether the dataset at hand belongs to the first group. If not, it verifies its membership to the second group and the third one after that. When the dataset does not belong to any of the first three groups, the method decides to which of the final five it belongs, which are mutually exclusive. Having done so, the weight vector in (6) is set according to the weighting scheme advised for the selected group and the main algorithm devised by the user can be run. Examples of this approach are provided in Section 5.4.

4.4. Explaining the observed behaviour

In this section, we answer some questions related to the observations made above and that naturally arise when studying the results in detail. These provide further insight in the performance of the different weighting schemes. We consider the following three challenges: a high number of classes (Section 4.4.1), a high number of instances (Section 4.4.2) and class imbalance (Section 4.4.3). Our analysis contains several important take-away messages for researchers who wish to use OWA-based fuzzy rough sets.

4.4.1. High number of classes

In this paper, we interpret a high number of classes as 'more than five'. This concerns all datasets in groups 4 and 5 by definition. Apart from these, the four datasets in group 2 all contain more than five classes as well. In group 1, datasets *flare* and *zoo* have six and seven classes respectively and the *derma* datasets from group 3 also contains six. In all, of the 50 datasets in Table 1, a subset of 16 have more than five classes. The average balanced accuracy of the weight settings over these datasets are 0.6641 (*Strict*), 0.5242 (*Add*), 0.6707 (*Exp*), 0.6597 (*Invadd*) and 0.6733 (*Mult*). *Strict* obtains the best result for six datasets, *Mult* and *Exp* each obtain the best result on four, while *Invadd* takes first place on the remaining two. *Add* has a poor result on 14 out of the 16 datasets. The other four alternatives each fail on two datasets. The following questions need to be addressed:

1. Why does *Add* not perform well when the number of classes is high?
2. In datasets with many classes, why are *Strict* and *Exp* the preferred settings?

We provide an answer to these two questions in the separate paragraphs below.

Question 1. Based on its mean balanced accuracy and the number of datasets on which it performs poorly, *Add* is highly inferior to the other weight settings (including *Strict*) when the number of classes is high. The reason for its bad performance is that there is too little difference in the sets of values to aggregate, that is, these sets have a high degree of overlap. Assume that there are six classes in the dataset (C_1 to C_6). In (6), the membership degree of x to $\underline{C_i}$ is computed by aggregating the values $1 - R(x, y)$ for instances y in any of the five other classes. This implies an overlap of four classes between the aggregation sets of $\underline{C_i}$ and $\underline{C_j}$. For example, $\underline{C_1}(x)$ and $\underline{C_2}(x)$ both use all values $1 - R(x, y)$ in classes C_3, C_4, C_5 and C_6. The only difference between the value vectors of $\underline{C_1}(x)$ and $\underline{C_2}(x)$ is that the former uses class C_2 and the latter class C_1.

The increased expected overlap between the value vectors in the lower approximation aggregations holds for the OWA model in general and is not specific for *Add*. Nevertheless, since *Add* assigns a large relative importance to all values (Section 3.3), the high overlap implies that the aggregated values for the different classes will be close together. Consequently, the ability to discern between classes decreases and prediction errors are made. Other weight settings are hindered less by the overlap, because their weight distribution is highly different to that of *Add* and places a clearer emphasis on the minimum (Section 3.3). On top of the overlap problem, a high number of classes also implies an increase in the size of the sets to aggregate. When the additive weight vector becomes longer, its behaviour approaches that of a regular average, which further accentuates the issue that the values $\underline{C_i}(x)$ are not sufficiently distinct.

Question 2. In our strategy presented in Section 4.3, we recommend *Strict* or *Exp* for datasets with more than five classes (groups 4 and 5). Table 2 shows that these are indeed the preferred configurations for such datasets in this study. Although they have been computed over a larger set of datasets, the average results listed at the beginning of this section confirm this.

Aside from the poor performance of *Add*, Table 2 also indicates that *Invadd* and *Mult* perform relatively less strongly on the datasets in groups 4 and 5. As explained in our answer to the previous question, when there are many classes in the dataset, there is a large overlap between the value vectors in the lower approximation computations. Although not to such an extent as *Add*, the *Invadd* and *Mult* settings also assign non-negligible weights to all values to be aggregated. As a result, they are also at risk for aggregated class approximation values that are too close to each other to adequately distinguish between them. *Mult* outperforms *Invadd*, because its weights are set up to decrease more rapidly going from right to left in the weight vector.

Strict and *Exp* are nearest neighbour approaches and only consider a small portion of the values in their aggregation step. This helps them to avoid the overlap problem and explains why they are the preferred weight setting in this situation. We make a further distinction between groups 4 and 5 based on the degree of imbalance in the dataset. The reason why *Strict* is preferred over *Exp* when the IR of a dataset is large is discussed in Section 4.4.3.

4.4.2. High number of instances

In this study, we put the threshold of a high number of instances to 4000. We realize that this is a small number in the big data era, but it is sufficiently large considering the characteristics described in Table 1. There are 13 datasets in our study with a size larger than 4000. These include the seven datasets from group 8, *marketing* and *satimage* from group 4, *abalone* and *winequal-w* from group 5, *nursery* from group 1 and *texture* from group 2. The average balanced accuracy of the weight settings on these datasets is 0.6594 (*Strict*), 0.5250 (*Add*), 0.6631 (*Exp*), 0.6132 (*Invadd*) and 0.6190 (*Mult*). *Strict* and *Exp* are clearly preferable in this case and the latter is the overall best choice.

The difference between *Strict* and *Exp* on the one hand and *Add, Invadd* and *Mult* on the other is that the nearest neighbour nature of the former two can cancel out the contribution of some instances (Section 3.3). For *Strict*, this is always the case, as it has zero weights for all but one position. For *Exp*, zero weights occur when the length of the vector increases, when due to its rapid (exponential) descent in weights from right to left in the vector W_L, only a small portion of values are assigned non-zero weights.

A larger dataset size implies a larger length of the weight vectors. The experimental results show that a nearest neighbour approach is more suitable for this situation than a full OWA aggregation, which takes all values into account. *Add, Invadd* and *Mult* lose some of their characteristics, because they insist on assigning some weight to all values. As the aggregation length becomes larger, important values (i.e. those close to the minimum) are assigned increasingly smaller weights to accommodate for this property, since OWA weights always sum to one (Definition 1). This is most prominently noticeable in *Add*, where the weight vector almost flattens out to a regular average.

The reason why *Exp* is preferred over *Strict* is the same as why k-nearest neighbour classification (kNN, with $k > 1$) is preferred over 1-nearest neighbour classification. It is more robust against noise and makes more confident predictions by relying on multiple near elements. Furthermore, weighted kNN is often favoured over uniform kNN, because the former assigns relatively more importance to nearer neighbours in its predictions [13].

4.4.3. Class imbalance

We have used the IR of a dataset on two occasions in our weight selection strategy: to make a distinction between groups 4 and 5 on the one hand and between groups 6 and 7 on the other. In this section, we explain why the weight choice can depend on the IR of a dataset. We answer two questions:

1. In datasets with a low number of classes and instances, why is *Add* the only good choice when the dataset is at least mildly imbalanced?
2. In datasets with many classes, why is *Strict* preferred in case of large imbalance and *Exp* in case of small to mild imbalance?

We discuss these two topics in separate paragraphs below.

Question 1. This question pertains to datasets with at most five classes, at most 4000 instances and an IR of at least two. The latter implies that the largest class is at least twice as large as the smallest class. In Table 3, five datasets have been assigned to group 7. Aside from these, *breast, car* and *splice* from group 1 and *spectfheart* from group 3 also have the properties listed above. The average balanced accuracy of the weight settings on these 9 datasets is 0.4852 (*Strict*), 0.5826 (*Add*), 0.5240 (*Exp*), 0.5577 (*Invadd*) and 0.5679 (*Mult*). The additive scheme attains the best average result and the highest number of wins (4 out of 9). Although *Add* appeared to be an inferior weight alternative in Section 4.2, it is clearly dominant on small, imbalanced datasets. We expect that the bad performance of *Strict* and *Exp* is due to their relation to the nearest neighbour classifier and the sensitivity of the latter to class imbalance.

Considering the results in more detail, we noticed that the other OWA alternatives often fall in the trap of class imbalance, that is, they assign instances to a majority class too easily. This implies a high accuracy for the majority class, but severely lower accuracies for minority classes. Our use of the balanced accuracy guarantees that a bad performance on small classes is not overshadowed by a good one on large classes. *Add* usually has similar accuracy rates for all classes in these datasets, reflected in its superior balanced accuracy values.

Expression (6) shows that the membership degree to the lower approximation of a class C is calculated by aggregating values based on elements that do not belong to C. Assume that the dataset contains two classes C_1 and C_2 and that the former is the majority class. Because of the above condition on the IR, this means that C_1 is at least twice as large as C_2. To classify an instance x, the predictor computes two values: $\underline{C_1}(x) = \text{OWA}_{W_L}(\{1 - R(x, y) \mid y \in C_2\})$ and $\underline{C_2}(x) = \text{OWA}_{W_L}(\{1 - R(x, y) \mid y \in C_1\})$. Due to the difference in class sizes, the first aggregation is taken over far less values than the second. The largest influence of this fact is felt by *Add*. As discussed above, the longer the additive weight vector becomes, the closer it is to a regular average. Since the aggregation lengths can be very different, the characteristics of *Add* on either class can severely vary as well. The longer aggregation ($\underline{C_2}(x)$) will be far closer to an average of its values than the shorter one ($\underline{C_1}(x)$). This difference in the treatment of classes is far less pronounced for the other OWA alternatives and in *Strict* it is even non-existent. Here lies the key to why *Add* is preferred in the presence of class imbalance: its high sensitivity to the length of the vector makes it process minority and majority classes very differently. It does not allow for majority elements to dominate the minority elements. The contributions of majority instances are almost averaged in the calculations, while those of minority elements are aggregated with a truer OWA aggregation. A similar conclusion holds when there are more than two classes.

In summary, like many other classifiers, the lower approximation predictor is sensitive to class imbalance. The additive weighting scheme inherently treats majority and minority classes differently, which is why it is the preferred choice here. We note that the work of [28] provides a more detailed study on appropriate OWA weight vectors when dealing with two-class imbalanced datasets.

Question 2. This question relates to datasets with more than five classes. We recommend to use *Exp* when the IR is at most 10 (group 4) and *Strict* otherwise (group 5). In Section 4.4.1, we have already explained why *Strict* and *Exp* are the best weight options for datasets with more than five classes. For datasets with a moderate IR, *Exp* is preferred over *Strict* for the same reason as given in Section 4.4.2, namely the higher prediction confidence and robustness when more than one near neighbour is used in the classification process. For a highly imbalanced dataset, the strict model is a better option. This is due to the class imbalance problem, which has a larger influence on kNN (with $k > 1$) than on 1NN. *Exp* loses some of its strength when the imbalance becomes large.

4.5. Remarks on our approach

As explained in Section 4.1, we have used the prediction performance of the OWA-based lower approximation to study the differences between the five weighting schemes. Machine learning methods using OWA-based fuzzy rough sets mostly rely on this operator (Section 2.3), which makes our study highly relevant for practical applications. We have clearly explained the effects of the weights in different situations. However, it is important to reflect on two aspects affecting the classifier performance: our selection of the five weighting schemes and similarity measure (5).

Evaluated weighting schemes. Additional weighting schemes can be found in the literature or devised by the reader. However, in light of the different properties exhibited by each scheme (Sections 3.1–3.3), we feel that we have made an appropriate selection. When the reader wishes to assess the adequacy of their custom weighting scheme within OWA-based fuzzy rough sets, they should be able to do so based on our discussion in Sections 4.3 and 4.4. We have based ourselves on the general defining characteristics of the schemes, in particular in Section 4.4, and our conclusions should carry over to other alternatives as well.

Instance similarity. We have chosen to fix the fuzzy relation measuring instance similarity to expression (5). This is a reasonable and intuitive similarity measure, which has been used in previous studies on fuzzy rough classifiers as well. Alternatives exist, but we do not expect that our observations and conclusions in Sections 4.3 and 4.4 will greatly change when a different (sensible) relation is used. When an alternative similarity measure is more suitable for a particular dataset, the performance of the classifier will improve, but we believe that the relative rankings of the weighting schemes will remain the same. Since we focus on the latter aspect, optimizing the similarity relation is of secondary importance and our default use of (5) is justified. Our conclusions in the previous sections are not strongly based on the instance similarity values. Naturally, like the nearest neighbour classifier, any fuzzy rough classifier may benefit from the application of metric learning [23]. A user that wishes to apply our simple classifier in a prediction task can opt to use our guidelines in conjunction with a data-dependent similarity measure derived with a metric learning technique.

5. Validation of the proposed strategy

We proceed with the validation of our proposal and do so in several steps:

1. Section 5.1: we first evaluate its performance on the 50 datasets in Table 1.
2. Section 5.2: next, we compare our manually detected trends with those extracted by a decision tree meta-learner.
3. Section 5.3: the third, important step is to validate the weight selection strategy on independent datasets that were not used in Section 4.
4. Section 5.4: finally, we consider two other applications aside from classification and show that our guidelines are useful in these settings as well.

We can analyse the difference in performance of our selection strategy and the fixed weight definitions by means of the Wilcoxon test [42]. This non-parametric test compares the results of methods M1 and M2 on n datasets. For each dataset, the difference in performance is computed by subtracting the result of M2 from that of M1. Afterwards, these differences are ranked from small to large in absolute value. The smallest difference is assigned rank 1, the largest rank n. R^+ and R^- are computed as the sum of the ranks of the positive and negative differences respectively. A higher value of R^+ is interpreted as a better performance of M1 compared to M2. A test statistic and p-value can be computed based on R^+ and R^-. When $R^+ > R^-$ and the p-value is smaller than the significance level α, it can be concluded that M1 performed significantly better than M2 on this group of datasets. In this paper, we use $\alpha = 0.05$.

5.1. Data from Table 1

If we follow our proposed strategy to select a weight setting for the 50 datasets in Table 1, the mean balanced accuracy increases to 0.7109. This is a noticeable increase compared to the highest value of 0.6891 in Section 4.2. On 25 out of the 50 datasets, our weight selection strategy chooses the weight setting with the best performance. On the 25 remaining ones, an alternative for which the balanced accuracy is at most 0.05 lower is chosen.

To verify whether the increase in balanced accuracy is statistically significant, we use our proposal as M1 in the Wilcoxon test and compare its performance to that of the five weight settings. The test shows that the proposed strategy outperforms every weight setting with statistical significance. In particular, using each of the five alternatives as M2, we find p-values of 0.000071 for *Strict* ($R^+ = 1048.5,\ R^- = 226.5$), 0.00000041 for *Add* ($R^+ = 1127.0,\ R^- = 98.0$), 0.000466 for *Exp* ($R^+ = 999.5,\ R^- = 275.5$), 0.002114 for *Invadd* ($R^+ = 921.0,\ R^- = 304.0$) and 0.002074 for *Mult* ($R^+ = 956.0,\ R^- = 319.0$). This good behaviour is not unexpected, as our strategy was derived based on the performance of the weight settings on these datasets. In Section 5.3, we validate our proposed strategy on independent datasets.

5.2. Decision tree

Our selection strategy in Section 4.3 has been constructed manually. We decided that any result that is within 0.05 of the best value for a dataset is acceptable. By doing so, we avoided overfitting our data and were able to construct sufficiently general and understandable weight selection rules. We did not aim to obtain the best overall result, but rather to avoid poor results.

Another option is to construct a meta-dataset, based on which the weight rules can be learned automatically. This dataset contains 50 entries, each corresponding to one dataset. We can use the data characteristics used in Section 4.3 (e.g. number of classes) as features. The class label of an entry is the name of the best performing weight setting for that dataset. To extract rules, a decision tree can be trained on the meta-dataset. The resulting tree model can be used to assign a weight setting to a new dataset based on its characteristics.

Although it removes the subjective aspect of our manual derivation, a limitation of this automated procedure is that there is no possibility to incorporate the same flexibility as before and accept slightly sub-optimal results. Using the meta-dataset corresponds to a different goal, namely to select the best weight setting as often as possible. The downside is that there is no safety net: if the best setting is not selected, a very poor alternative may very well be chosen.

We have used the rpart function from the rpart package in R [30] to train a decision tree on the meta-dataset. All parameters were set to their default settings, apart from ‘minbucket’, which we set to four. For this value, the tree has eight leaf nodes. We deemed this appropriate, because we used eight groups of datasets as well. As splitting features, the tree can select the number of instances, the number of classes, the IR and whether or not there are only nominal features in a dataset. Fig. 2 presents the resulting decision tree.

The tree detects the same general trends as we did. Its first split is made on the number of classes. We interpreted up to five classes as a small number, but the tree considers any number larger than two as high. An explanation may be that many of our datasets consist of only two classes (24 out of 50).

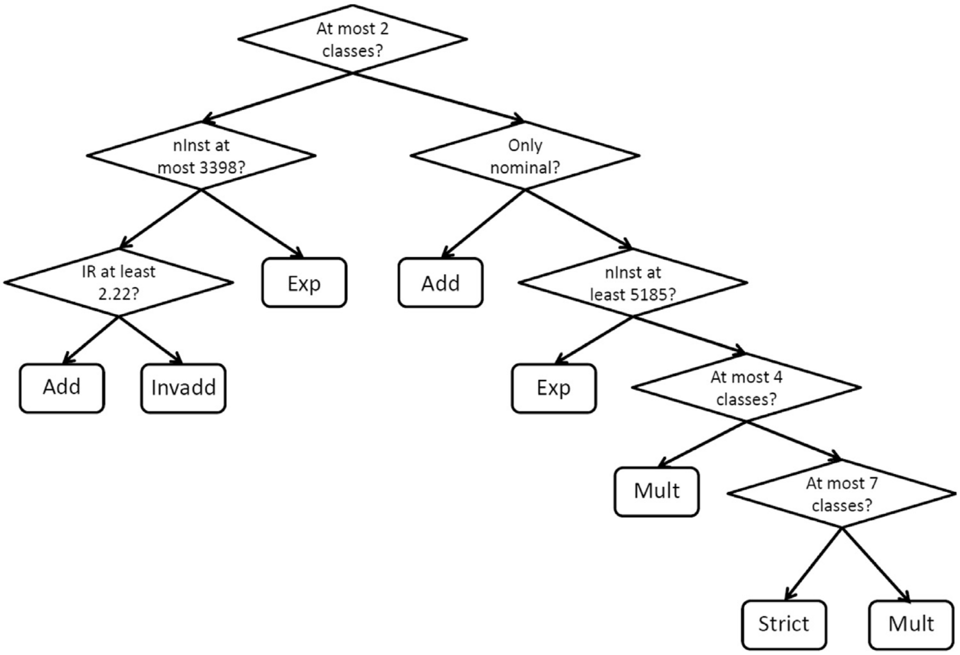


Fig. 2. Decision tree learned from the meta-dataset. For each internal node, the left child fulfils the condition, the right child does not.

In datasets with a small number of classes and many instances, *Exp* should be used. We modelled this as well and explained this behaviour in Section 4.4.2. When both the number of classes and instances are small, *Invadd* is used when the classes are more or less balanced and *Add* when they are not. This roughly corresponds to groups 6 and 7 in Section 4.3. We used the threshold 2 to decide when the imbalance is too large, the tree uses 2.22.

For datasets with only nominal features, the tree selects the additive scheme. Table 2 shows that *Add* indeed has the most wins on these datasets, but gives a bad result on some of them as well. We compromised and selected *Mult*. The decision tree cannot do so, because it focuses on the best performing settings.

In datasets with a higher number of classes, we made a division based on the IR. This does not happen in the decision tree. It groups all datasets with more than two classes in the right sub-tree, while we only considered the subset of datasets with more than five classes. The tree again advises the use of *Exp* when the dataset is large (see Section 4.4.2). Among the remaining datasets, a distinction is only made based on the number of classes. *Mult* is used for datasets with three, four or at least seven classes and *Strict* when there are five or six. This division seems somewhat artificial and cannot be easily explained. The tree is probably overfitting the meta-dataset.

The mean balanced accuracy on the datasets in Table 1 using the weight settings provided by the tree is 0.7089 and the best scheme is selected for 27 datasets. In Section 5.1, our weight scheme selection strategy selected the best setting on 25 datasets and attained a mean balanced accuracy of 0.7109. Although it leads to fewer wins, its power to compromise allows our proposal to perform better on average. Furthermore, it never selects a poor performing scheme on the 50 datasets, while the tree does so on four of them. Aside from their performance, our strategy can be favoured over the decision tree for a second reason: it is easier to understand and explain.

5.3. Independent data

In this section, we evaluate the performance of our proposed strategy on independent datasets, that is, datasets that have not been used in this study thus far. This evaluation will further reinforce the demonstrated efficacy and validity of our proposal. The 20 datasets used in this section are listed in Table 4 and are representatives of the eight groups defined in Section 4.3. They were obtained from the KEEL, UCI and Weka platforms. These datasets and their partitions can be downloaded from http://www.cwi.ugent.be/sarah.php.

We present the balanced accuracy values of this evaluation in Table 5. Apart from the results obtained using our proposed strategy, the table also shows the performance of the five individual weight settings. As before, the best value for each dataset is shown in bold typeface, while any value that is more than 0.05 lower than this optimum is underlined. The advantages of our proposal are clear. We obtain the highest balanced accuracy on average, the most wins and the fewest poor results. The table also shows that our selection strategy is not infallible either, as it does not perform well on the *mushroom* dataset. The *Mult* setting is selected, because this dataset contains only nominal features. However, if we were

Table 4
Description of the 20 independent datasets used in Section 5.3 We list the number of features (nFeat), the number of instances (nInst), the number of classes (nCl) and the IR. Together with the number of features, we specify whether they are all nominal (Y) or not (N).

Name	nFeat	nInst	nCl(IR)	Name	nFeat	nInst	nCl(IR)
appendicitis	7(N)	106	2(4.05)	iris	4(N)	150	3(1)
banknote	4(N)	1372	2(1.25)	letter	16(N)	20,000	26(1.11)
biodeg	40(N)	1055	2(1.96)	magic	10(N)	19,020	2(1.84)
credit	15(N)	653	2(1.21)	messidor	19(N)	1151	2(1.13)
ctg	21(N)	2126	10(10.92)	mushroom	22(Y)	5644	2(1.62)
eye_detection	14(N)	14,980	2(1.23)	optdigits	64(N)	5620	10(1.03)
faults	33(N)	1941	2(1.88)	penbased	16(N)	10,992	10(1.08)
grub	8(N)	155	4(2.58)	seismic	18(N)	2584	2(14.2)
hepatitis	19(N)	80	2(5.15)	sensor	24(N)	5456	4(6.72)
housevotes	16(Y)	232	2(1.15)	transfusion	4(N)	748	2(3.2)

Table 5
Balanced accuracy results of the classification by the OWA-based fuzzy rough lower approximation operator on independent datasets. With the results of our weight selection strategy, we list the selected weight setting between brackets.

Dataset	Strict	Add	Exp	Invadd	Mult	Proposal
appendicitis	0.7514	**0.7938**	0.7479	0.7868	0.7542	**0.7938** (A)
banknote	**0.9987**	0.9247	**0.9987**	0.9961	**0.9987**	0.9961 (I)
biodeg	0.8153	0.7139	0.8381	0.8216	**0.8513**	0.8216 (I)
credit	0.8194	0.8695	0.8582	**0.8704**	0.8571	**0.8704** (I)
ctg	**0.7379**	0.3999	0.7301	0.6283	0.6793	**0.7379** (S)
eye_detection	0.8456	0.5903	**0.8615**	0.8147	0.7226	**0.8615** (E)
faults	0.9897	0.6748	**0.9919**	0.9611	0.9904	0.9611 (I)
grub	0.2638	**0.3963**	0.3075	0.3638	0.3500	**0.3963** (A)
hepatitis	0.8184	0.8232	0.8199	**0.8633**	0.7715	0.8232 (A)
housevotes	0.9137	0.9017	**0.9209**	0.9125	0.9125	0.9125 (M)
iris	0.9333	**0.9533**	0.9400	**0.9533**	**0.9533**	0.9333 (S)
letter	0.9428	0.6124	**0.9632**	0.9556	0.9537	**0.9632** (E)
magic	0.8129	0.7606	**0.8384**	0.8126	0.7907	**0.8384** (E)
messidor	0.6282	0.6056	0.6534	0.6546	**0.6638**	0.6546 (I)
mushroom	0.9593	0.8026	**0.9749**	0.9713	0.9034	0.9034 (M)
optdigits	0.9843	0.9094	**0.9857**	0.9798	0.9834	**0.9857** (E)
penbased	0.9936	0.7624	**0.9939**	0.9671	0.9902	**0.9939** (E)
seismic	0.5514	**0.7015**	0.5299	0.5872	0.4998	**0.7015** (A)
sensor	0.9224	0.7393	**0.9255**	0.9024	0.9080	**0.9255** (E)
transfusion	0.5708	0.6528	0.6048	**0.6745**	0.6047	0.6528 (A)
Mean	0.8126	0.7294	0.8242	0.8238	0.8069	**0.8363**
# best/poor	2/4	4/12	10/3	4/2	4/6	11/1

to ignore this particular guideline, *mushroom* would be assigned to group 8, because it has 2 classes and 5644 instances. In this case, it would be processed with *Exp*, which is the preferred setting for this dataset.

We compare the performance of our proposal to the five weight settings using a Wilcoxon test. We can conclude that it performs significantly better than *Strict* ($R^+ = 167.5,\ R^- = 42.5,\ p = 0.01821$), *Add* ($R^+ = 178.0,\ R^- = 12.0,\ p = 0.00027$) and *Mult* ($R^+ = 165.5,\ R^- = 44.5,\ p = 0.02272$). We cannot conclude that our approach provides a significantly better result than *Invadd* ($R^+ = 144.5,\ R^- = 65.5,\ p = 0.14827$) or *Exp* ($R^+ = 122.5,\ R^- = 87.5,\ p = 0.47034$), although the higher values for R^+ do indicate a preference in our favour. Table 5 also demonstrates that a higher balanced accuracy is obtained on average in this case, as well as a higher number of wins and lower number of poor results. We would also like to stress that since our selection strategy is based on simple data characteristics, the computation time to assess which weighting scheme should be selected based on our instructions is negligible.

The aim of this paper has been to provide a better understanding of the impact of the different weighting schemes on the OWA-based fuzzy rough set model and advise how to make an appropriate choice between them. Our objective has not been to develop a new state-of-the-art classifier. Nevertheless, we briefly remark that the use of our guidelines lifts the classification result of the simple fuzzy rough lower approximation classifier to the level of the random forest method [8]. This is a far more complex model and is generally accepted as a strong classifier. Using the default random forest method from the Weka platform [16] with ten decision trees, the mean balanced accuracy on the datasets from Table 4 is 0.8373. No significant differences between this result and the last column from Table 5 are detected, indicating that our basic fuzzy rough classifier with a sensible weight selection can yield competitive results with such a popular classifier as the random forest method.

5.4. Other applications

As a final validation step, we use our guidelines within the OWA-FRPS instance selection method [33] and the POSNN classifier [32,34]. Both use the fuzzy rough positive region $POS(\cdot)$. For an instance x, $POS(x)$ is computed as its membership degree the OWA-based lower approximation of its own class. In [32], the use of *Invadd* was advised for both methods, regardless of the dataset on which they are applied. As observed in Section 4.2, this weighting scheme can appear a good default choice based on its average performance, but does not necessarily perform well on all datasets. We now evaluate whether using our guidelines instead benefits the performance of OWA-FRPS and POSNN. As explained in Section 4.3, this is achieved by verifying to which of the eight groups a dataset belongs and setting the weighting scheme used within (6) accordingly. We use the independent datasets listed in Table 4 and provide the full results on http://www.cwi.ugent.be/sarah.php.

The OWA-FRPS method computes the quality of all training instances as their membership degree to the positive region and derives a threshold above which the quality is deemed sufficiently high. Only instances with a quality exceeding this threshold are retained in the dataset. We combine it with the 1NN classifier, as done in [32,33]. When we fix the weighting scheme to *Invadd*, the average reduction in the number of instances is 27.63% and the balanced accuracy of 1NN after OWA-FRPS is 0.8097. Using our guidelines results in an average reduction of 30.42% and a balanced accuracy of 0.8133. This provides three advantages: (i) we relieve the user from setting the OWA weighting scheme, when they are not fully comfortable with using a default option, (ii) on average, the datasets are reduced slightly more and (iii) the classification performance of the posterior classifier is maintained, even somewhat improved (albeit not statistically significantly).

The POSNN classifier is a weighted extension of the fuzzy nearest neighbour classifier of [21]. The contribution of each neighbour is weighted by its membership degree to the OWA-based positive region. Note that the lower approximation operator is not used as a predictor, as done in the construction of our guidelines, but rather as a weighting mechanism. We have set the number of neighbours to 10. The use of *Invadd*, as recommended in [32] leads to an average balanced accuracy of 0.8084 on the datasets in Table 4, while using our guidelines increases this value to 0.8101. According to the results of the Wilcoxon test, this improvement is significant ($R^+ = 148.5,\ R^- = 41.5,\ p = 0.029774$).

6. Conclusion

Fuzzy rough set theory is a mathematical tool to model uncertainty in real-world data. It has been used in several machine learning domains, but a limitation of the traditional model is its high sensitivity to noise. Several noise-tolerant fuzzy rough set models have been proposed in the literature. Among them, the OWA-based fuzzy rough set model generalizes the strict minimum and maximum in the traditional approximation operators to OWA aggregations.

The OWA-based fuzzy rough approximation operators are easy to interpret and implement and offer a robustness in the presence of noise that is clearly demonstrated in previous work. The OWA-based model has been shown to outperform the state-of-the-art in various machine learning domains. Nevertheless, OWA-based fuzzy rough set approaches have not experienced the same widespread adoption or popularity as the traditional fuzzy rough set model. We believe that further advances can be made in many areas by exploiting the strengths and interpretability of this versatile and flexible model.

In this paper, we have provided a crucial step on the way to increase the popularity of OWA-based fuzzy rough set theory. Its approximation operators depend on the user-defined aggregation weight setting. Up until now, no clear set of rules has existed for a suitable weighting scheme selection such that users of this fuzzy rough set model were left to their own devices. Through systematic and rigorous comparison of five different weighting schemes, we have remedied this shortcoming and provided a clear strategy for the weight selection process. The efficacy of the strategy is further supported and validated by an evaluation on independent datasets and in different applications. Our contribution is twofold. Most importantly, we have provided the community with clear weighting scheme selection guidelines for the OWA-based lower approximation, an operator repeatedly used in machine learning methods based on this fuzzy rough set model. Secondly, we have also offered further insight into the definition and inner workings of OWA-based fuzzy rough sets. No such work has been previously presented.

We feel that we have succeeded in our aim to facilitate the use of OWA-based fuzzy rough set theory by providing an easy-to-follow weight selection strategy. Aside from the weight definition, the user does not need to set any parameter. We have seen the benefit of this fuzzy rough set model in our own research and are confident that this work will contribute to a more wide-spread adoption of OWA-based approaches, on top of the ones included in this paper.

To compare the different weighting schemes, we have focused on the classification prediction of the lower approximation operator. A user interested in the OWA-based upper approximation can nevertheless use our explanations on the behaviour of the different settings as baseline information in their weight selection process.

Conflict of interest

None.

Acknowledgements

The research of Sarah Vluymans is funded by the Special Research Fund (BOF) of Ghent University. Yvan Saeys is an ISAC Marylou Ingram Scholar. The research of Chris Cornelis is funded by the Odysseus programme of the Science Foundation-Flanders (FWO).

References

[1] M. Amiri, R. Jensen, Missing data imputation using fuzzy-rough methods, Neurocomputing 205 (2016) 152–164.
[2] M. Amiri, R. Jensen, M. Eftekhari, N. Mac Parthaláin, Dataset condensation using owa fuzzy-rough set-based nearest neighbor classifier, in: Proceedings of the 2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2016), IEEE, 2016, pp. 1934–1941.
[3] S. An, Q. Hu, W. Pedrycz, P. Zhu, E. Tsang, Data-distribution-aware fuzzy rough set model and its application to robust classification, IEEE Trans. Cybern. 46 (12) (2016) 3073–3085.
[4] G. Beliakov, How to build aggregation operators from data, Int. J. Intell. Syst. 18 (8) (2003) 903–923.
[5] G. Beliakov, R. Mesiar, L. Valaskova, Fitting generated aggregation operators to empirical data, Int. J. Uncertain. Fuzziness Knowl.-Based Syst. 12 (2) (2004) 219–236.
[6] T. Boongoen, Q. Shen, Clus-DOWA: a new dependent OWA operator, in: Proceedings of the 2008 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2008), IEEE, 2008, pp. 1057–1063.
[7] T. Boongoen, Q. Shen, Nearest-neighbor guided evaluation of data reliability and its applications, IEEE Trans. Syst. Man. Cybern.Part B 40 (6) (2010) 1622–1633.
[8] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.
[9] C. Cornelis, M. De Cock, A. Radzikowska, Vaguely quantified rough sets, in: Proceedings of the International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing, Springer, 2007, pp. 87–94.
[10] C. Cornelis, N. Verbiest, R. Jensen, Ordered weighted average based fuzzy rough sets, in: Proceedings of the 5th International Conference on Rough Sets and Knowledge Technology (RSKT 2010), Springer, 2010, pp. 78–85.
[11] L. D'eer, N. Verbiest, C. Cornelis, L. Godo, A comprehensive study of implicator–conjunctor-based and noise-tolerant fuzzy rough sets: definitions, properties and robustness analysis, Fuzzy Sets Syst. 275 (2015) 1–38.
[12] D. Dubois, H. Prade, Rough fuzzy sets and fuzzy rough sets, Int. J. Gen. Syst. 17 (2–3) (1990) 191–209.
[13] S. Dudani, The distance-weighted k-nearest-neighbor rule, IEEE Trans. Syst. Man. Cybern. SMC-6 (4) (1976) 325–327.
[14] D. Filev, R. Yager, On the issue of obtaining OWA operator weights, Fuzzy Sets Syst. 94 (2) (1998) 157–169.
[15] R. Fullér, P. Majlender, An analytic approach for obtaining maximal entropy OWA operator weights, Fuzzy Sets Syst. 124 (1) (2001) 53–57.
[16] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. Witten, The WEKA data mining software: an update, ACM SIGKDD Explor. Newsl. 11 (1) (2009) 10–18.
[17] T. Ho, M. Basu, M. Law, Measures of geometrical complexity in classification problems, in: Data Complexity in Pattern Recognition, Springer, 2006, pp. 1–23.
[18] Q. Hu, L. Zhang, S. An, D. Zhang, D. Yu, On robust fuzzy rough set models, IEEE Trans. Fuzzy Syst. 20 (4) (2012) 636–651.
[19] R. Jensen, C. Cornelis, Fuzzy-rough nearest neighbour classification, Trans. Rough Sets XIII (2011) 56–72.
[20] R. Jensen, Q. Shen, New approaches to fuzzy-rough feature selection, IEEE Trans. Fuzzy Syst. 17 (4) (2009) 824–838.
[21] J. Keller, M. Gray, J. Givens, A fuzzy k-nearest neighbor algorithm, IEEE Trans. Systems Man Cybern. SMC-15 (4) (1985) 580–585.
[22] G. Klir, B. Yuan, Fuzzy Sets and Fuzzy Logic, 4, Prentice hall New Jersey, 1995.
[23] B. Kulis, Metric learning: a survey, Found. Trends Mach. Learn. 5 (4) (2013) 287–364.
[24] C. Lemke, M. Budka, B. Gabrys, Metalearning: a survey of trends and technologies, Artif. Intell. Rev. 44 (1) (2015) 117–130.
[25] M. O'Hagan, Aggregating template or rule antecedents in real-time expert systems with fuzzy set logic, in: Proceedings of the 22nd Asilomar Conference on Signals, Systems and Computers, 2, IEEE, 1988, pp. 681–689.
[26] Z. Pawlak, Rough sets, Int. J. Comput. Inf. Sci. 11 (5) (1982) 341–356.
[27] A. Radzikowska, E. Kerre, A comparative study of fuzzy rough sets, Fuzzy Sets Syst. 126 (2) (2002) 137–155.
[28] E. Ramentol, S. Vluymans, N. Verbiest, Y. Caballero, R. Bello, C. Cornelis, F. Herrera, IFROWANN: imbalanced fuzzy-rough ordered weighted average nearest neighbor classification, IEEE Trans. Fuzzy Syst. 23 (5) (2015) 1622–1637.
[29] Y. Sun, A. Wong, M. Kamel, Classification of imbalanced data: a review, Int. J. Pattern Recognit.Artif. Intell. 23 (04) (2009) 687–719.
[30] T. Therneau, B. Atkinson, B. Ripley, The rpart package, 2010.
[31] V. Torra, On the learning of weights in some aggregation operators: the weighted mean and OWA operators, Mathw.Soft Comput. 6 (2/3) (1999) 249–265.
[32] N. Verbiest, Fuzzy Rough and Evolutionary Approaches to Instance Selection, Ghent University, 2014 Ph.D. thesis.
[33] N. Verbiest, C. Cornelis, F. Herrera, OWA-FRPS: a prototype selection method based on ordered weighted average fuzzy rough set theory, in: Proceedings of the International Workshop on Rough Sets, Fuzzy Sets, Data Mining and Granular-Soft Computing, Springer, 2013, pp. 180–190.
[34] N. Verbiest, C. Cornelis, R. Jensen, Fuzzy rough positive region based nearest neighbour classification, in: Proceedings of the 2012 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2012), IEEE, 2012, pp. 1–7.
[35] N. Verbiest, C. Cornelis, R. Jensen, Quality, frequency and similarity based fuzzy nearest neighbor classification, in: Proceedings of the 2013 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2013), IEEE, 2013, pp. 1–8.
[36] S. Vluymans, C. Cornelis, F. Herrera, Y. Saeys, Multi-label classification using a fuzzy rough neighborhood consensus, Inf. Sci. 433 (2018) 96–114.
[37] S. Vluymans, L. Deer, Y. Saeys, C. Cornelis, Applications of fuzzy rough set theory in machine learning: a survey, Fundam. Inf. 142 (1–4) (2015) 53–86.
[38] S. Vluymans, A. Fernández, Y. Saeys, C. Cornelis, F. Herrera, Dynamic affinity-based classification of multi-class imbalanced data with one-versus-one decomposition: a fuzzy rough set approach, Knowl. Inf. Syst. 56 (1) (2018) 55–84.
[39] S. Vluymans, N. Mac Parthaláin, C. Cornelis, Y. Saeys, Fuzzy rough sets for self-labelling: an exploratory analysis, in: Proceedings of the 2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2016), IEEE, 2016, pp. 931–938.
[40] S. Vluymans, D. Sánchez Tarragó, Y. Saeys, C. Cornelis, F. Herrera, Fuzzy multi-instance classifiers, IEEE Trans. Fuzzy Syst. 24 (6) (2016) 1395–1409.
[41] S. Vluymans, D. Sánchez Tarragó, Y. Saeys, C. Cornelis, F. Herrera, Fuzzy rough classifiers for class imbalanced multi-instance data, Pattern Recognit. 53 (2016) 36–45.
[42] F. Wilcoxon, Individual comparisons by ranking methods, Biom. Bull. 1 (6) (1945) 80–83.
[43] Z. Xu, Dependent OWA operators, in: Proceedings of the International Conference on Modeling Decisions for Artificial Intelligence, Springer, 2006, pp. 172–178.
[44] R. Yager, On ordered weighted averaging aggregation operators in multicriteria decisionmaking, IEEE Trans. Syst. Man. Cybern. 18 (1) (1988) 183–190.
[45] R. Yager, D. Filev, Induced ordered weighted averaging operators, IEEE Trans. Syst. Man. Cybern.Part B 29 (2) (1999) 141–150.
[46] R. Yager, J. Kacprzyk, The Ordered Weighted Averaging Operators: Theory and Applications, Springer Science & Business Media, 2012.
[47] L. Zadeh, Fuzzy sets, Inf. Control 8 (3) (1965) 338–353.
[48] S. Zhao, E. Tsang, D. Chen, The model of fuzzy variable precision rough sets, IEEE Trans. Fuzzy Syst. 17 (2) (2009) 451–467.