Information Sciences 260 (2014) 98–119

Contents lists available at ScienceDirect

Information Sciences

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

Fuzzy nearest neighbor algorithms: Taxonomy, experimental analysis and prospects

Joaquín Derrac [a,b,*], Salvador García [c], Francisco Herrera [b,d]

[a] School of Computer Science & Informatics, Cardiff University, Cardiff CF24 3AA, United Kingdom
[b] Department of Computer Science and Artificial Intelligence, University of Granada, 18071 Granada, Spain
[c] Department of Computer Science, University of Jaén, 23071 Jaén, Spain
[d] Faculty of Computing and Information Technology – North Jeddah, King Abdulaziz University, 21589 Jeddah, Saudi Arabia

ARTICLE INFO

Article history:
Received 4 December 2012
Received in revised form 23 August 2013
Accepted 27 October 2013
Available online 13 November 2013

Keyword:
Fuzzy nearest neighbor
Nearest neighbor
Taxonomy
Experimental framework
Classification

ABSTRACT

In recent years, many nearest neighbor algorithms based on fuzzy sets theory have been developed. These methods form a field, known as fuzzy nearest neighbor classification, which is the source of many proposals for the enhancement of the k nearest neighbor classifier. Fuzzy sets theory and several extensions, including fuzzy rough sets, intuitionistic fuzzy sets, type-2 fuzzy sets and possibilistic theory are the foundations of these hybrid techniques, designed to tackle some of the drawbacks of the nearest neighbor rule.

In this paper the most relevant approaches to fuzzy nearest neighbor classification are reviewed, as are applications and theoretical works. Several descriptive properties are defined to build a full taxonomy, which should be useful as a future reference for new developments. An experimental framework, including implementations of the methods, datasets, and a suggestion of a statistical methodology for results assessment is provided. A case of study is included, featuring a comparison of the best techniques with several state of the art crisp nearest neighbor classifiers. The work concludes with the suggestion of some open challenges and ways to improve fuzzy nearest neighbor classification as a machine learning technique.

1. Introduction

The nearest neighbor (NN) rule is a nonparametric method for pattern classification [44] based on instances [1]. Introduced by Fix and Hodges in 1951 [28], the NN rule gained considerable popularity after 1967, when some of its formal properties were described by Cover and Hart [23]. Cover's work was a milestone in a subject which has since become a lively research field for many researchers in pattern recognition and machine learning [4,100] and the study and development of one of the top ten algorithms in data mining [101].

Although the NN rule has been introduced in many research problems, its foremost application belongs to supervised classification, in which patterns contained in a test set *TS* are classified using the patterns included in a training set *TR* as reference. Here, a pattern x follows the usual definition $\mathbf{x} = \{x_1, x_2, \ldots, x_d, \omega\}$, where d is the number of attributes that describe the data and ω is its assigned class.

The general definition of the NN rule in supervised classification, the k nearest neighbors classifier (k-NN), considers the use of the most similar (nearest) k patterns in *TR* to derive the class of a test pattern. More formally, let $\mathbf{x}_i$ be a training

* Corresponding author at: Department of Computer Science and Artificial Intelligence, University of Granada, 18071 Granada, Spain. Tel.: +34 958 240598; fax: +34 958 243317.

E-mail addresses: jderrac@decsai.ugr.es (J. Derrac), sglopez@ujaen.es (S. García), herrera@decsai.ugr.es (F. Herrera).

0020-0255/$ - see front matter
http://dx.doi.org/10.1016/j.ins.2013.10.038

pattern from $TR, 1 \leqslant i \leqslant N$ (where N is the number of patterns in TR) and $\mathbf{x}_j$ be a test pattern from $TS, 1 \leqslant j \leqslant M$ (where M is the number of patterns in TS). During the training process, the k-NN classifier simply stores the true class ω of each training pattern $\mathbf{x}_i$. In test phase, the decision rule predicts a class $\hat{\omega}$ for the test pattern $\mathbf{x}_j$, according to the true class ω of the majority of its k nearest neighbors (its most similar patterns from TR). In the case of a tie, $\hat{\omega}$ is given by the closest nearest neighbor that belongs to one of the tied classes.

Despite its simplicity, the k-NN classifier has been widely studied from many perspectives, pursuing the improvement of its classification accuracy or mitigating some of its well-known shortcomings. The most critical of its shortcomings are the necessity of storing the full training set when performing the classification task (in contrast to most machine learning procedures, which only require a model to be stored); the relatively low efficiency of the computation of the decision rule (due to the necessity of computing the similarity of the test pattern with every pattern of TR); the low tolerance to noise of the classifier (especially when k is set to $k = 1$) and the fact that the k-NN classifier relies exclusively on the existing data, assuming that the training set defines the decision boundaries among the classes perfectly, which is not always the case.

The aforementioned drawbacks have been analyzed extensively by the research community. As a result, many approaches have been proposed regarding, for example, the computation of similarity measures [16], the optimum choice of the k parameter [60], the definition of weighting schemes for patterns and attributes [96,51], the adaptation of the algorithm to data [43], the development of fast and approximate versions of the NN rule, devised to quicken the computation of the nearest neighbors [37,74,5,70], and the reduction of the training data [31,91,22,26].

Fuzzy Sets Theory (FST) [107] has been the basis of a remarkable number of these approaches. In the context of nearest neighbor classification, FST allows imprecise knowledge (such as the membership of outliers to any of the classes of the problem) to be represented and fuzzy measures to be introduced, which provide, for example, an enhanced way of describing the similarities between the instances that represent a problem.

These components, managed in a crisp way by the original k-NN classifier, are usually the focus of the extensions proposed by fuzzy nearest neighbor algorithms (together with the automatic set up of the k parameter and the definition of new ways of combining the votes of the nearest neighbors). In the literature, their study has been tackled considering FST, various extensions and related approaches including fuzzy rough sets [24], intuitionistic fuzzy sets [7], possibilistic theory [108] and type-2 fuzzy sets [58].

Supported by the former approaches, the development of the field has heightened since 1983 and 1985, with the first works in the area published by Jóźwik [56] and Keller et al. [59]. New approaches regarding both improvements of the k-NN model and applications to real-world problems have been proposed, drawing the attention of many researchers and practitioners.

All these approaches have been proposed with a clear objective: improving the accuracy of the NN rule. By introducing soft memberships (to represent those instances which are not typical prototypes of each class), improved similarity measures (to adapt the way in which distances are computed to the fuzzy memberships), new decision rules (to incorporate both the memberships and the distances to the final classification of the test instances), the precision of the classifier is expected to be enhanced over that of the original NN rule. Also, some fuzzy nearest neighbor algorithms exhibit other interesting capabilities such as not requiring a special set-up of the k parameter.

In this work we present a study of the current status of fuzzy nearest neighbor classification. A survey of methods is provided, focused on the ways in which the NN rule has been extended and modified throughout these years. A full taxonomy is proposed, considering the different techniques involved in the development and description of the new proposals. This taxonomy is founded on several distinctive traits identified among the most relevant methods.

Moreover, a full experimental framework, including a set of well-known publicly accessible and representative supervised classification problems, is offered. This framework offers implementations of the essential methods for reference, and suggests a statistical methodology based on nonparametric procedures, which should be sufficient to provide a rigorous confirmation of the differences reported in most cases. The framework's description is concluded with a case study comparing fuzzy NN based methods with a set of representative crisp NN based approaches. After the analysis of results (including a further analysis of the relative performance of the methods as the number of instances, attributes and classes of the problems grow), the paper concludes with the suggestion of several interesting research trends that remain open within the topic.

We also have developed a website with complementary material to the paper http://sci2s.ugr.es/fuzzyKNN/survey.php including detailed descriptions of the algorithms analyzed and all of the material used in our case study (implementations, data sets and partitions). Full results of the experimental study can also be found there.

The rest of this work is organized as follows: Section 2 presents a survey of the existing literature, reviewing the most interesting proposals published based on the fuzzy NN rule. Fuzzy nearest neighbor algorithms are then characterized, with respect to several distinctive traits. Section 3 shows the taxonomy proposed, based on common characteristics shared among the methods. Section 4 describes our experimental framework, including data sets, algorithms and statistical procedures. Section 5 shows the experimental study performed. Section 6 discusses several open problems as a way of suggesting the future development of the field. Finally, Section 7 concludes the paper.

2. Fuzzy nearest neighbor algorithms

Many proposals have been presented since the publication of the first works in the field. These proposals focus not only on improvements to the classical model, but also other topics such as the use of different extensions of fuzzy sets, the addition of a preprocessing mechanism based on data reduction, or the development of real-world applications, describing instances of problems tackled successfully by fuzzy nearest neighbor techniques.

This section is devoted to surveying relevant works in these directions, describing the key elements of each approach. A more detailed description of the methods can be found at http://sci2s.ugr.es/fuzzyKNN/survey.php.

After the survey, several common properties of the methods are identified and described. These properties are used to characterize the main approaches surveyed, providing with an insight into the existing differences in the design of the methods.

2.1. A survey on fuzzy nearest neighbor classification

Since the presentation of the very first proposals, fuzzy nearest neighbor classification has become a distinctive area within the field of nearest neighbor classification and instance based learning. The addition of FST based mechanisms to the traditional approaches has enabled very accurate and flexible classification models to be defined, with outstanding results when applied to supervised learning problems.

Through this section, both classical approaches and new extensions will be surveyed, including proposals based on possibilistic theory, intuitionistic sets, fuzzy rough sets and data preprocessing. A description of other interesting proposals using both nearest neighbor classification and fuzzy sets is also included. The survey is finished with several remarkable examples of applications of fuzzy nearest neighbor classification to real-world scenarios.

2.1.1. Nearest neighbor algorithms based on fuzzy sets theory

The first proposal of a fuzzy nearest neighbor classifier was presented by Jóẃik [56] in 1983. This classifier, JFKNN, is an improved version of the standard *k*-NN. It is based on a learning scheme of class memberships, providing each training instance with membership array which defines its fuzzy membership to each class. After the learning process, the final classification is performed similarly to *k*-NN, but every neighbor uses its membership array for the voting rule, instead of just giving one vote as in the crisp *k*-NN.

Two years later, Keller et al. [59] proposed what has since become the major reference in this field (with currently more than 450 citations in the ISI Web of Science). FuzzyKNN, the classifier described in this work, has been the baseline of many advanced methods hybridizing FST and *k*-NN classifiers. Furthermore, there are plenty of applications in many fields of research based on this model, mainly due to its good behavior when tackling supervised learning problems.

FuzzyKNN introduced two modifications to the original *k*-NN rule:

- A preliminary training phase is introduced. In this phase, class memberships are derived for each training instance, obtaining a value in [0, 1] for each instance and class. Although Keller proposed three different methods for computing these memberships[1] the best performing method requires for each instance $\mathbf{x}_i$ to compute the *kInit* nearest neighbors in the training data.[2] Then, memberships are assigned following Eq. (1)

$$u_c(\mathbf{x}_i) = \begin{cases} 0.51 + (v_c/kInit) * 0.49 & \text{if } c = \omega \\ (v_c/kInit) * 0.49 & \text{otherwise.} \end{cases} \tag{1}$$

where v_c are the number of neighbors found belonging to class c, and ω is the class of $\mathbf{x}_i$ in the original data.
The effect of Eq. (1) is that instances close to the center of the classes kept their original crisp memberships[3] but instances close to the boundaries between classes spread half of their membership among the neighbors' classes. However, it is interesting to note that the 0.51 and 0.49 coefficients still ensure that the largest membership will be assigned to the ω class, regardless of the neighboring instances.[4] Also, the sum of the memberships to all classes will always be 1.
- A modified voting rule in which each neighboring instance votes for every class, using the memberships learned during the training phase. These votes are weighted according to the inverse of the distance to the instance to be classified, and finally all votes are added. The final class, $\hat{\omega}$, is obtained as the class with the greatest combined vote.

In addition to FuzzyKNN, Keller's work also presented FuzzyNPC, which is a prototypical version of FuzzyKNN. It works by using only one prototype per class (which is obtained as the mean of every instance of the class in the training data), obtaining $\hat{\omega}$ using the inverse of the distances computed to each prototype. Hence, this classifier becomes a faster (but less accurate) version of FuzzyKNN.

---

[1] One of them is the 'crisp' one: to assign a membership of 1 to the class of the instance in the original data, and 0 to the rest of the classes.
[2] *kInit* is usually set to a value between (3, 10).
[3] 1.0 to their original class ω and 0.0 to the rest.
[4] 0.51 or more to their original class ω and 0.49 or less to the rest.

Another classical way of designing fuzzy nearest neighbor classifiers is the use of clustering algorithms to estimate the membership values of each training instance. In [8], Bedzek and Chuah proposed a fuzzy version of ISODATA to perform this task for the *k*-NN classifier. Later, in [9], the Fuzzy C-Means clustering algorithm was introduced to obtain the memberships. Béreau and Dubuisson [11] also presented a clustering algorithm for this task, but aiming to minimize the entropy between classes, instead of maximizing the accuracy of the underlying *k*-NN classifier.

All of these classical approaches are reviewed by Yang and Chen [103], whose work also includes a theoretical proof that the FuzzyKNN rule is bounded above by twice the Bayes risk, extending the results of Cover and Hart for the *k*-NN rule [23]. The convergence properties of this error are also studied in [104], extending the original review.

Considering as a starting point these classic approaches, many extensions were developed introducing new schemes of computation of the weights, new ways of calculating the distances, and other ways of improving the nearest neighbor classifiers by using fuzzy sets.

The most common approach is the modification of the way in which membership weights are computed. In [40], the VWFKNN classifier sets weights according to the standard deviation of the neighbors' class membership values. In this way, weights can model a discriminant function identifying the different classes of the classification problem. Another example is [81], in which Pham designed a method based on a kriging system to obtain the membership weights.

Another trend of research is focused on the modification of the computation of distances. In [62] the distances between the instances are modified depending on its typicalness. Using expert knowledge (expert council interviewing for a medical problem, in this case) fuzzy decision rules are derived in order to obtain an accurate nearest neighbor classifier. Fuzzy distances, represented by fuzzy numbers, are also considered in [75]. In this second case, the introduction of fuzzy distances allows Mitchell and Schaefer method to adapt automatically the value of *k* according to the local density of the training instances.

Other extensions are focused in the enhancement of the FuzzyKNN by the optimization of the *kInit* and *m* parameters. For example, GAFuzzyKNN [46] employs a genetic algorithm to optimize both values. A parallel implementation of a genetic algorithm designed for this task is also presented in [85].

2.1.2. Interval type-2 fuzzy sets based approach

Type-2 fuzzy sets have been the basis of a fuzzy nearest neighbor approach, presented in [20]. In that work, the IT2FKNN classifier is proposed as an alternative way of discarding the necessity of setting up the parameter *kInit* in the original definition of FuzzyKNN. This is achieved by introducing interval type-2 fuzzy sets to represent the memberships computed by considering distinct choices of the parameter *kInit*. Type-2 fuzzy sets are built considering all the different memberships computed, and then a type reduction operation is performed to obtain a final, combined value, representative of all the choices initially considered. The rest of the phases of the algorithm are similar to the original definition of FuzzyKNN.

2.1.3. Possibilistic k-NN methods

Possibilistic classification extends fuzzy classification in the sense that the set of memberships assigned to every instance is not constrained; that is, in most of the fuzzy classification approaches the sum of the membership degree to every class of each instance must be 1 (see, for example, Eq. (1) for FuzzyKNN). This property does not hold in possibilistic classification, which means that non-canonical situations can be represented using this paradigm: Using a possibilistic model an instance could belong to two classes simultaneously (that is, it might have a degree of membership of 1 in more than one class), or could not be a clear representative of any class (having a sum of memberships much lower than 1).

D-SKNN [25] is the first example implementing this model. It is a *k*-NN classifier based on the Dempster–Shafer theory, and incorporates mechanisms to manage uncertainty and reject unclear instances. Another related model has recently been proposed in [27], incorporating lower previsions as generic models for uncertainty management.

Possibilistic instance based learning is also analyzed in [49]. The paper is focused on the development of a theoretical possibilistic framework, linking its properties with those of nearest neighbor classification and analyzing advanced concepts concerning uncertainty in nearest neighbor classification, similarity measures, noise and outliers detection, and incomplete information management. It also presents a classifier based on these concepts, PosIBL, which does not need the specification of the *k* parameter of the classic *k*-NN rule.

2.1.4. Intuitionistic k-NN methods

Intuitionistic fuzzy sets have also been used to develop fuzzy nearest neighbor classifiers. By incorporating the concept of nonmembership, it is possible to model some additional situations in an effort to characterize the classification problems as accurately as possible.

In [38] the IFSKNN classifier was proposed. In this algorithm, a value of membership is computed for each instance, as the distance to the mean of the class. Then, the nonmembership value is computed in a similar way, as the distance to the nearest mean of the rest of the classes. This enables typical instances to be represented with a high degree of membership, whereas noisy instances will be assigned with a high degree of non-membership.[5] The classification is completed using memberships and nonmemberships to modify the distances computed by a *k*-NN classifier.

[5] Note that with this representation, outliers – instances which are far from all the classes – will be represented with very low degrees of membership and nonmembership, thus the degree of indeterminateness can be used as a way of representing outliers in the training data.

A second approach using intuitionistic fuzzy sets was presented in [63]. IF-KNN considers the nonmembership degree to be the opposite of the membership of each instance to the class, and employs both values to determine the contribution of each neighbor's vote to the final classification.

Finally, in [39] the IFV-NP classifier was proposed. It is a prototypical version of IFSKNN, in which, after obtaining the prototypes, a procedure is carried out to adjust the degrees of membership and nonmembership in accordance with the distances to the center of the classes.

2.1.5. Preprocessing approaches via data reduction

Preprocessing methods have become an effective way of enhancing the performance of general nearest neighbor classifiers. Among them, prototype selection [31] and prototype generation [91] fields have inspired the first preprocessing methods for fuzzy nearest neighbor classifiers.

Regarding prototype selection, the FENN classifier [105] is based on Wilson's editing rule for *k*-NN [98]: All instances in the training set are checked and those whose classification by the FuzzyKNN rule does not agree with its original class are removed. CFKNN [109] is also inspired by a classic method, Hart's condensing rule [41], although it uses the sample fuzzy entropy to determine whether an instance is finally removed or kept. Another representative example is the PFKNN method [6], which first builds a set of prototypes representing the border points of different clusters in the data and then adds to this reference set those instances which could be misclassified. The algorithm concludes with a pruning phase in which non-relevant prototypes are discarded.

Finally, it is also possible to find prototype generation methods such as, for example, the Gayar et al. method [35], which describes the use of Fuzzy C-Means to obtain the membership weights of prototypes generated in an iterative way.

2.1.6. Fuzzy rough sets based approaches

Recently, several approaches to nearest neighbor classification based on fuzzy rough sets have been proposed. Most of them aim to improve the quality of the classification performed with the combined support of the rough sets and fuzzy sets theories.

A first proposal, FRNNA, was presented in [12]. This classifier incorporates the lower and upper approximations of the memberships to the decision rule, in an effort to deal with both fuzzy uncertainties and rough uncertainties. A second proposal – FRNN [86] – develops this aspect further, associating fuzzy uncertainties with the existing overlapping between classes and rough uncertainties with the lack of a proper number of features to describe the data. Another main feature of this method is that it does not require a fixed *k* value for the classification rule.

Fuzzy-rough nearest neighbor classification is developed in [53]. In these works, the FRNN-FRS and FRNN-VQRS classifiers are described. They employ fuzzy rough sets and vaguely quantified rough sets, respectively. The first classifier is presented as an improvement of FRNN, whereas in the second one vaguely quantified rough sets are introduced to reduce the sensitivity of the classifier to noise. Finally, a further step in fuzzy-rough nearest neighbor classification is presented in [82], where Qu et al. presents an approach to hybridizing kernel-based classification with fuzzy rough sets.

2.1.7. Further extensions

In addition to the wide range of proposals that have appeared in the literature, presenting a rich variety of fuzzy nearest neighbor classifiers, the joint use of fuzzy sets and the nearest neighbor classifier has further enhanced work in this area. Several of such works have focused either on the application of fuzzy nearest neighbor rules to tackle different problems (other than classification) or on other ways of combining FST and *k*-NN. This subsection surveys some of the most interesting approaches:

- The success of FuzzyKNN and other fuzzy nearest neighbor classifiers has inspired similar techniques used in incremental data problems [89] (when the full training set is not available at the training phase), outliers detection in temporal series [78], regression [84], semi-supervised learning for monitoring evolving systems [42], multi-label text categorization [54] or low quality data problems [71].
- FST has become an interesting tool for the enhancement of the classic *k*-NN classifier throughout data preprocessing approaches. Some notable examples include [50,76] in which an evolutionary instance selection method is presented and extended. The method is enhanced through the transformation of the instances into circular-conic fuzzy rules, which are finally used to train the classifiers. Instance selection is also the focus of [52,57], in which two different instance selection methods based on fuzzy rough sets are described.

  In addition to preprocessing approaches, several works have also investigated further ways of extending the nearest neighbor classifiers. For example, in [102], a theoretical description of a lineal programming method is provided. This method is aimed at the design of ordered weighted average operators for the decision rule of *k*-NN. A different approach is presented in [66], which includes an approximated nearest neighbor classifier [5] as a fast classification model, based on fuzzy rough sets.
- Finally, FuzzyKNN has also been considered as a part of larger and more complex classification algorithms such as [106], in which a boosting approach including FuzzyKNN, evolutionary feature selection and decision trees is presented. Another example is [36], in which a genetic algorithm is used to optimize a one-versus-all ensemble of FuzzyKNN classifiers. In

[19], Chua and Tan proposed an hybrid algorithm including a genetic fuzzy system, FuzzyKNN and a weighting scheme for the distance function of the classifier. Also, in [14], a model integrating FuzzyKNN and several multi layer neural networks as the core of a Mamdani type fuzzy inference system was proposed.

2.1.8. Applications

Fuzzy nearest neighbor classifiers have been selected by many practitioners in very different fields of science and industry. Among them, FuzzyKNN stands out as the preferred choice from among a remarkable number of applications. The amount of proposals describing very specific modifications of the original classifiers, designed to tackle the difficulties that arise in each problem, is also worthy of note.

The first application was presented by Cabello et al. [13] in which the Fuzzy C-Means clustering algorithm and FuzzyKNN are used together to tackle a problem of arrhythmia detection. Other recent applications are [17], in which diabetes diseases are diagnosed by incorporating FuzzyKNN into a full artificial immune recognition system, and [73], in which the joint use of particle swarm optimization, principal component analysis and FuzzyKNN is proposed for a thyroid disease diagnosis problem.

Other medical technologies have also benefited from the use of fuzzy nearest neighbor classifiers: Liao et al. [67–69] presented several approaches to classifying radiographic images, including the use of feature extraction, Fuzzy C-Means clustering and FuzzyKNN. Another notable example is [65], in which Leszczynski et al. analyze the performance of FuzzyKNN with different classic distance measures (euclidean, mahalanobis, etc.) in a framework of decision making in radiotherapy.

Another major field of application is bioinformatics. Many approaches to protein identification and prediction includes FuzzyKNN [48,93,45], some of which incorporate additional mechanisms, such as [61], in which a parallel implementation of FuzzyKNN is suggested.

Outside of the medical and bioinformatics fields there are also plenty of applications selecting FuzzyKNN as a suitable classifier (for example, [83] developing a wine classification system, [47] in which FuzzyKNN is used to classify web documents, and [92] in which a computer vision approach to duck meat color classification is presented).

Moreover, it is also easy to find other applications in which FuzzyKNN is combined with preprocessing techniques (for example, [55] using Fuzzy C-Means for preprocessing data describing a cellular manufacturing system, or [64] which includes principal component analysis to reduce the dimensionality of data in a mold detection problem) or with other general methods (such as the recent proposals of [15] for bankruptcy prediction incorporating FuzzyKNN in a particle swarm optimization scheme, or [18] combining the output of several FuzzyKNN classifiers in a human action recognition problem.

Finally, there are not many applications including advanced fuzzy nearest neighbor classifiers, although [110,80] are among the most remarkable. In the former work, Zhu and Basir presented a classifier inspired by D-SKNN, incorporating a fast implementation scheme, and applied it to image classification problems. In the latter, Petridis and Kaburlasos designed a k-NN method based on fuzzy interval numbers for the prediction of sugar production throughout different years.

2.2. Common properties of fuzzy nearest neighbor algorithms

Many different characteristics govern the behavior of the different fuzzy nearest neighbor algorithms that have appeared in the literature. However, there are several common traits of major importance, from the point of view of nearest neighbor based classification, which are worthy of analysis:

- Membership degree to a class: In the crisp definition of the NN rule, training patterns are restricted to belonging to a single class, regardless of their spatial properties. Allowing fuzzy memberships (i.e. replacing ω with a membership function representing the pattern's assignment to two or more classes) can be very useful for modeling many difficult (but common) situations in supervised classification, such as uncertain knowledge about the true class of a pattern (e.g. due to the presence of noise).
- Similarity measure: The usage of similarities between patterns as a way of fuzzyfying the contribution of each neighbor to the decision process may allow an enhancement of the discriminative power of the training data, thus improving the classification performance.
- Decision rule: In the k-NN classifier, the final decision about the $\hat{\omega}$ class of a test pattern is given by a single majority voting process. Other decision rules may be derived to combine the votes of the nearest neighbors, providing the classifier with new ways of assigning the $\hat{\omega}$ class of the test pattern.

These traits can be categorized into 3 groups: *Membership*, *Distance* and *Voting*. Each of the techniques analyzed will only show one trait of each category. A last category, *Others*, includes additional traits that may belong or not to any technique. A description of each category and trait is given as follows:

- *Membership*: This category refers to the way in which the instance's memberships to each class of the problem is represented. Four different schemes are considered:
  - Crisp scheme: Classical crisp memberships are used; that is, instance's memberships are considered to be 1 in the particular class to which the instance belongs, and 0 in the rest.

– Fuzzy scheme: A fuzzy set defines the instance membership to each class. In this case, the sum of the memberships of an instance to all the classes of the problem will always be 1. No other restrictions are imposed, although it is very common for methods using this scheme to assign a higher membership degree to the class to which the instance belongs in the initial training data.
– Possibilistic scheme: In this extension of the fuzzy scheme, the requirement of having the sum of all memberships equal to 1 is removed. Usually, class membership degrees remain normalized in the $[0,1]$ interval, but here there is no objection to representing an instance with full membership to several classes or without belonging to any class at all.
– Intuitionistic scheme: When intuitionistic sets are used, two values (in $[0,1]$) are used to represent each instance membership (membership and non-membership). Both values are simultaneously managed by the algorithm decision rule during the classification process.

- *Distance*: This category refers to the way in which the distances computed between the instance to be classified and each training instance are considered:
  – Inverse weight: The most common approach is to use the inverse of the similarity value computed (usually the Euclidean distance between the test and the training instance) as a weight to increase the strength of the neighbor vote in the decision rule.
  – Distance modulation: Different schemes can be applied to incorporate the distances computed to the decision rule, modifiying its effect through the use of additional procedures such as kernels or exponential relations.
  – Not used: Some of the techniques analyzed do not consider the absolute value of similarity computed in the decision rule. Although they use distances to find the nearest neighbors of the test instance, these values are disregarded as soon as the neighbors have been found.
- *Voting*: The definition of the voting rule used by the classifier. Typically, an additive scheme is chosen, which means that votes emitted by each neighboring instance (possibly weighted by its relative distance to the test instance) are added to create the final output of the classifier. However, different voting schemes may be selected:
  – Classical: An additive scheme is used to combine the neighbors' votes.
  – Global: Every instance in the training set is considered in the voting process (instead of just the neighboring instances).
  – Best neighbor: Only the best neighbor found among the k nearest ones (not necessarily the nearest) is used to determine the output of the classifier.
- *Others*: In this category, other relevant traits of nearest neighbor classifiers are included:
  – Independence of k: The method does not require a value of k to be set for the decision rule.
  – Preprocessing: In addition to the classification process, this technique also performs some form of data preprocessing. Thus, as a side effect, the original training data is usually reduced (for example, by means of a prototype selection or generation technique). Note, however, that the main objective of the method remains the classification task.
  – Center based: The classification is oriented to relating test instances with the class whose center is nearest. This effect – desirable for many classification problems, although it may be harmful in certain cases – is typical of those techniques which rely on a clustering procedure to analyze the training data.

Table 1 displays a summary of such characteristics, highlighting which fuzzy nearest neighbor algorithms share them.

In each row, a check mark (✔) is shown for each specific capability possessed by the respective algorithm. Algorithms are denoted either by their acronym[6] or by their author's name. The algorithm's main reference is also provided.

3. Taxonomy

By considering the traits described in the former section, it is possible to detail a general categorization of the fuzzy nearest neighbor algorithms. Fig. 1 proposes a taxonomy founded both on the general field on which each technique is based and on some of the traits analyzed previously.

The first level of the taxonomy is devoted to describing each technique depending on its main category: Fuzzy sets, type-2 fuzzy sets, possibilistic methods, intuitionistic fuzzy sets, fuzzy rough sets and preprocessing approaches via data reduction. Among these categories, a second and a third level is introduced to discriminate between methods belonging to the same field:

- For fuzzy sets based approaches, the main differential characteristics are independence from the k value and the usage of distances to weight the computation of the votes in the decision rule.
- Possibilistic methods are also categorized depending on whether they are dependent to the set up of the k value or not.
- Similarly to the latter, intuitionistic fuzzy sets based methods are also categorized by their dependence on the set up of the k value. Intuitionistic methods dependent on k can be further characterized as center based methods, focused on determining the center of each of the classes of the problem and adapting their classification scheme to the centers found.

[6] These algorithms are part of the experimental framework that will be presented below; their full name will be provided in Table 3.

Table 1
Common characteristics of fuzzy nearest neighbor algorithms.

Acronym/ name	Ref.	Member.				Distance			Voting			Others		
		Crisp	Fuzzy	Posibilistic	Intuitionistic	Inverse weight	Distance modulation	Not used	Classical	Global	Best neighbor	Independence of *k*	Preprocessing	Center based
JFKNN	[56]		✓					✓		✓		✓		
FuzzyKNN	[59]		✓			✓			✓					
FuzzyNPC	[59]	✓				✓					✓	✓	✓	✓
FCMKNN	[9]		✓					✓	✓					✓
Kissiov et al.	[62]		✓					✓	✓					
D-SKNN	[25]			✓			✓		✓					
IFSKNN	[38]				✓			✓	✓					✓
IF-KNN	[63]				✓			✓	✓					
FENN	[105]		✓			✓			✓				✓	
VWFuzzyKNN	[40]		✓			✓			✓					
IFV-NP	[39]				✓		✓		✓			✓	✓	✓
Mitchell et al.	[75]	✓					✓			✓		✓		
IT2FKNN	[20]		✓			✓			✓					
PosIBL	[49]			✓			✓			✓		✓		
FRKNNA	[12]		✓					✓	✓					
Pham et al.	[81]		✓			✓			✓					
Gayar et al.	[35]		✓			✓					✓	✓	✓	✓
GAFuzzyKNN	[46]		✓			✓			✓					
FRNN	[86]			✓			✓			✓		✓		
PFKNN	[6]		✓			✓			✓				✓	
CFKNN	[109]		✓			✓			✓				✓	
FRNN-FRS	[53]	✓					✓				✓			
FRNN-VQRS	[53]	✓					✓				✓			
Qu et al.	[82]	✓					✓				✓			
Desterke et al.	[27]			✓			✓		✓					

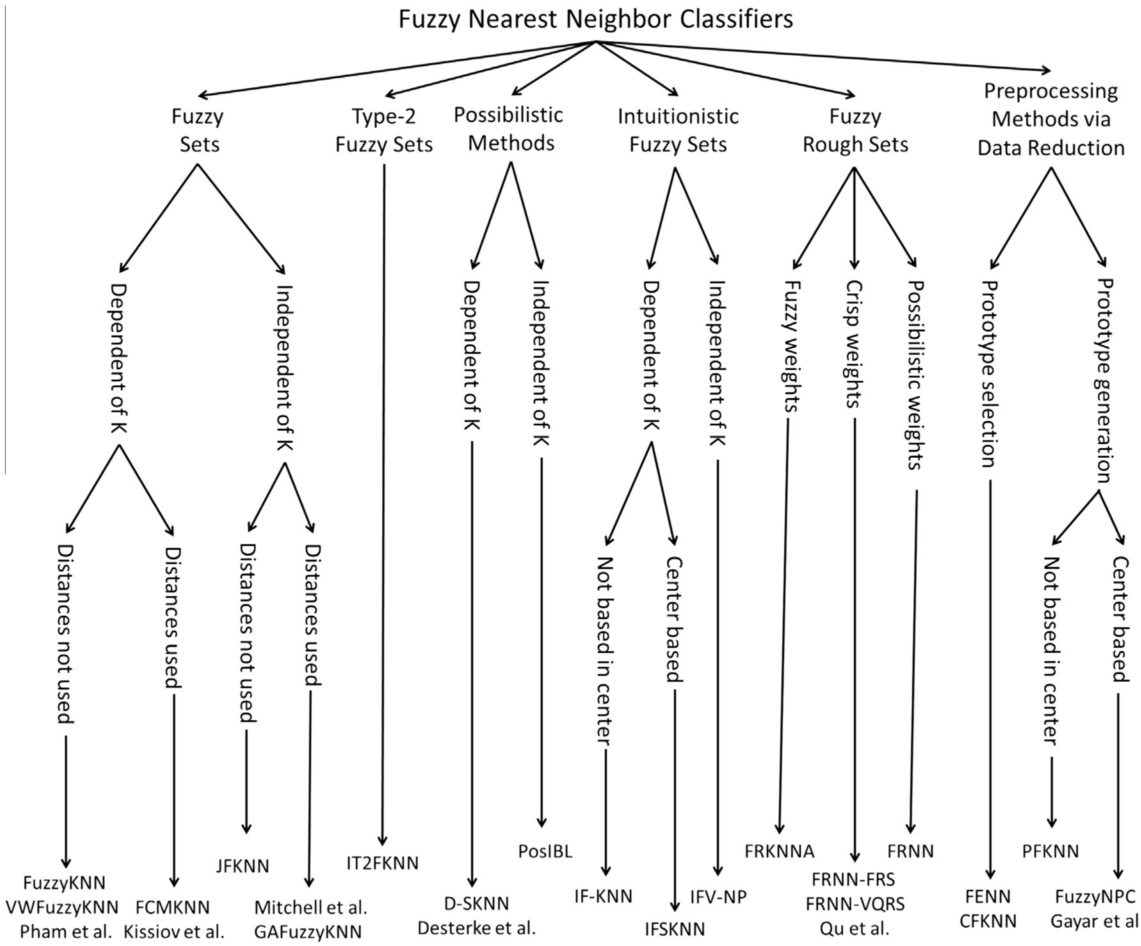

Fig. 1. Proposed taxonomy of fuzzy nearest neighbor classifiers. Each method is to be categorized into one of the six major classes: Fuzzy sets, type-2 fuzzy sets possibilistic methods, intuitionistic fuzzy sets, fuzzy rough sets or preprocessing approaches via data reduction. These classes are further divided according to some key properties of the classifiers, including dependence on the *k* parameter, the use of distances to weight the votes and other relevant traits.

- Fuzzy rough sets based methods can be characterized according to the scheme used to represent the membership of the training instances to the classes of the problem: They can use either fuzzy weights, possibilistic weights or crisp weights.
- The natural way of classifying preprocessing based approaches is to refer to the preprocessing field on which they are based (prototype selection or prototype generation). In addition, some prototype generation based techniques can be categorized as center based methods.

The properties displayed in this taxonomy may be very helpful in understanding how an specific algorithm works, also enabling the inclusion of new algorithms developed in the future. Although different schemes could have been chosen here, the different levels established ensure that any technique (already analyzed or new) can easily be placed in one of the major categories, using the second and the rest of levels to refine its categorization as much as necessary.

4. Experimental framework for fuzzy nearest neighbor classifiers

A critical step in the analysis of computational intelligence methods is the testing of their behavior in a controlled environment. In the context of supervised classification, this requires the consideration of several elements including problems instances, comparison methods and evaluation tools.

In this section, we present the experimental framework developed in order to analyze the most representative fuzzy nearest neighbor classifiers of the state of the art. It will provide useful material for characterizing the current status of the field, facilitating the experimental comparisons required in future developments.

The elements included in the framework are the following:

Table 2
Data sets included in the framework.

Data set	#Ins.	#At.	#Cl.	Data set	#Ins.	#At.	#Cl.
Appendicitis	106	7	2	Penbased	10992	16	10
Balance	625	4	3	Phoneme	5404	5	2
Banana	5300	2	2	Pima	768	8	2
Bands	539	19	2	Ring	7400	20	2
Bupa	345	6	2	Satimage	6435	36	7
Cleveland	297	13	5	Segment	2310	19	7
Dermatology	358	34	6	Sonar	208	60	2
Ecoli	336	7	8	Spambase	4597	57	2
Glass	214	9	7	Spectfheart	267	44	2
Haberman	306	3	2	Tae	151	5	3
Hayes–Roth	160	4	3	Texture	5500	40	11
Heart	270	13	2	Thyroid	7200	21	3
Hepatitis	80	19	2	Titanic	2201	3	2
Ionosphere	351	33	2	Twonorm	7400	20	2
Iris	150	4	3	Vehicle	946	18	4
Led7Digit	500	7	10	Vowel	990	13	11
Mammographic	830	5	2	Wdbc	569	30	2
Marketing	6876	13	9	Wine	178	13	3
Monk-2	432	6	2	Winequality-red	1599	11	11
Movement	360	90	15	Winequality-white	4898	11	11
New Thyroid	215	5	3	Wisconsin	683	9	2
Page-blocks	5472	10	5	Yeast	1484	8	10

- *Data sets*: A large set of 44 well-known supervised classification data sets is provided, and their main characteristics are described.
- *Fuzzy nearest neighbor classifiers*: The framework features a library including the most relevant fuzzy nearest neighbor classifiers in the state of the art.
- *Comparison algorithms*: A collection of several representative crisp nearest neighbor classifiers is presented. They will be considered in order to test the behavior of the best performing fuzzy nearest neighbor classifiers in a more general scenario.
- *Parameters configuration*: The guidelines followed to configure the parameter of each method are described. Also, the set up considered for the k parameter is discussed in depth, given its importance with respect to most of the methods.
- *Performance measures*: Several performance measures have been considered for use in analyzing the behavior of the methods. Their characteristics are described, as well as the motivations for including them in the experimental study.
- *Statistical procedures*: Several hypothesis testing procedures are considered to determine whether the differences found in the experimental study between the performance of multiple algorithms are significant or not.

All of the contents of this framework are publicly available in http://sci2s.ugr.es/fuzzyKNN/framework.php. They are described in depth throughout the rest of this section.

4.1. Data sets

The framework includes 44 supervised classification data sets. This is a compilation of well-known problems in the area, taken from the KEEL-dataset repository[7] [2] and the UCI repository [29].

Table 2 summarizes the main characteristics of the data sets. For each one, the table provides its number of instances (*#Ins.*), attributes (*#At.*) and classes (*#Cl.*).

The data sets considered are partitioned following a ten folds cross-validation procedure, and their values are normalized in the interval $[0,1]$ to equalize the influence of attributes with different range domains.

By using the ten folds cross-validation procedure [90], each data set is randomly partitioned into ten subsets, preserving the same size (the same number of instances) and the same class distribution between partitions. When running the classifier, an iterative process is followed where one partition is selected as the test set and the training set is composed of the rest. This process is continued until every partition has served as the test set once. Then, the final results per dataset are obtained by averaging the results obtained over the ten partitions.

Note that no data set includes nominal values, and instances with missing values have been discarded. As will be discussed later, this is a limitation of the methods in the current state of the art: Nominal and missing values are often neglected by most of the existing fuzzy nearest neighbor classifiers.

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

4.2. Fuzzy nearest neighbor classifiers

The library of fuzzy nearest neighbor classifiers features 19 different methods. All of them have been coded in Java, under the guidelines of the KEEL project [3]. They have been coded considering all the instructions provided by the authors in their respective papers.

Table 3 lists the methods included. For each one we provide its **acronym**, **name**, **year** of publication, and the main reference (**Ref.**) describing the work. We consider that this selection properly represents the current state of the art in the area. Note that we have not considered those approaches whose description in their original work was incomplete, or whose use requires additional resources (such as [62], in which expert human knowledge is required prior to running the algorithm).

4.3. Comparison algorithms

In addition to the full library of fuzzy nearest neighbor classifiers, we have added a set of representative crisp nearest neighbor classifiers to our study. Its inclusion in the study will allow the behavior of the fuzzy nearest neighbor classifiers to be tested in a more general environment, considering a wider range of methods. The crisp nearest neighbor classifiers chosen are described as follows:

- *k-NN classifier (k-NN)*: The performance of the *k*-NN classifier will be studied as a reference for the rest of methods [23].
- *Edited Nearest Neighbors (ENN)*: A prototype selection algorithm based on the edition of noisy instances. Instances whose class do not match their nearest neighbors' class are removed from the training set. After the edition process, the *k*-NN classifier is used to obtain the final classification [98].
- *Integrated Decremental Instance-Based Learning algorithm (IDIBL)*: An integrated model featuring instance selection, selection of kernel function for the voting process, and automatic determination of the *k* value and other related parameters [99].
- *Adaptive k nearest neighbors classifier (KNNAdaptive)*: A modification of the distance measure of the NN rule. Distances in this method are divided by the distance of the reference prototype to its nearest enemy (the nearest prototype from a different class) [95].
- *k Symmetrical nearest neighbors classifier (KSNN)*: A modification of the voting rule of the NN classifier, where votes are considered for those instances in which the test instance would be one of its *k* nearest neighbors [77].

Table 3
List of methods included in the framework.

Acronym	Name	Year	Ref.
JFKNN	Jóźwik Fuzzy *k*-Nearest Neighbor algorithm	1983	[56]
FuzzyKNN	Fuzzy *k*-Nearest-Nearest Neighbors classifier	1985	[59]
FuzzyNPC	Fuzzy Nearest Prototype classifier	1985	[59]
FCMKNN	Fuzzy C-Means *k*-Nearest Neighbors classifier	1986	[9]
D-SKNN	Dempster–Shafer theory based *k*-Nearest Neighbors classifier	1995	[25]
IFSKNN	Intuitionistic Fuzzy Sets *k*-Nearest Neighbors classifier	1995	[38]
IF-KNN	Intuitionistic Fuzzy *k*-Nearest Neighbors classifier	1995	[63]
FENN	Fuzzy Edited Nearest Neighbor classifier	1998	[105]
VWFuzzyKNN	Variance Weighted Fuzzy *k*-Nearest Neighbors classifier	1999	[40]
IFV-NP	Intuitionistic Fuzzy Version of *k*-Nearest Neighbors classifier	2000	[39]
IT2FKNN	Interval Type-2 Fuzzy *k*-Nearest Neighbors classifier	2003	[20]
PosIBL	Possibilistic Instance Based Learning	2003	[49]
FRKNNA	Fuzzy Rough *k*-Nearest Neighbors Approach	2003	[12]
GAFuzzyKNN	Genetic Algorithm for Fuzzy *k*-Nearest Neighbors classifier	2005	[46]
FRNN	Fuzzy-Rough Nearest Neighbor algorithm	2007	[86]
PFKNN	Pruned Fuzzy *k*-Nearest Neighbors classifier	2010	[6]
FRNN-FRS	Fuzzy-Rough Nearest Neighbor classifier – Fuzzy Rough Sets	2011	[53]
FRNN-VQRS	Fuzzy-Rough Nearest Neighbor classifier – Vaguely Quantified Rough Sets	2011	[53]
CFKNN	Condensed Fuzzy *k*-Nearest Neighbors classifier	2011	[109]

- *Nearest Subclass Classifier (NSC)*: An application of the minimum variance clustering method to the generation of prototypes for the NN rule [94].
- *Prototype Weighting algorithm (PW)*: A gradient descent based algorithm developed for computing prototype weights to minimize the *leave one out* error of the NN rule over the training set [79].

4.4. Parameter configuration

An essential factor in the set up of the experimental study is the configuration of the different parameters that governs the behavior of each method. In the majority of cases, the experiments focus their attention on the k parameter, highlighting the best value for each method or testing different values. In this study, given the wide range of approaches considered and the variability between author's recommendations in each work, we have chosen to take a representative set of fixed values for the k parameter, $k \in \{3, 5, 7, 9\}$.

Accordingly, $k = 1$ is excluded since, as with most of classical nearest neighbor approaches, the majority of fuzzy nearest neighbor algorithms become the 1-NN rule when a single neighbor is considered, regardless of the additional fuzzy-based mechanisms incorporated. In addition, no further values of k beyond $k = 9$ are considered. This is due to the smoothing nature of the k parameter, which, if increased by too much, may render the discriminative capabilities of most of the nearest neighbor classification algorithms powerless, degenerating into a majority classifier. In fact, most of the experimental studies in nearest neighbor classification follow this rationale, sticking to some of the k values defined above.

The rest of the configuration parameters are fixed to the values recommended by the respective authors (the similarity function considered is the Euclidean one). For the sake of fairness, in those cases where k does not need to be chosen (either because it is determined automatically or because it is not necessary to choose a value), a similar number of configurations has been considered, tuning other specific parameters according to the author's recommendations.

All the methods included in the experiments, both fuzzy and crisp nearest neighbor classifiers, will follow these parameter configuration rules.

4.5. Performance measures

Several performance measures can be considered in the analysis of the different algorithms of the study. In this case, accuracy and kappa are considered as precision measures, whereas running time is chosen to measure the efficiency of the methods in general terms.

Accuracy is defined as the number of successful hits relative to the total number of classifications. It has been by far the most commonly used metric for assessing the performance of classifiers for years [100,4]. Cohen's kappa [21] is an alternative to the accuracy rate, a method, known for decades, that compensates for random hits in the same way as the AUC measure [10]. Kappa can be computed using the following expression:

$$kappa = \frac{N\sum_{i=1}^{c} x_{ii} - \sum_{i=1}^{c} x_{i.}x_{.i}}{N^2 - \sum_{i=1}^{c} x_{i.}x_{.i}} \quad (2)$$

where x_{ii} is the cell count in the main diagonal of the classification confusion matrix, N is the number of examples, c is the number of class values, and $x_{.i}, x_{i.}$ are the columns' and rows' total counts, respectively. Kappa ranges from -1 (total disagreement) through 0 (random classification) to 1 (perfect agreement). For multi-class problems, it is a very useful, yet simple, metric for measuring the accuracy of the classifier while compensating for random successes.

Finally, average running time per partition is considered as a way of measuring the differences between methods with respect to computational cost. Its usage will allow us to determine which methods require a greater amount of time to complete the classification tasks.

In our study, the running time will measure the time spent from the point at which the training and test sets have been loaded into memory and preprocessed, to the point at which the output file reporting the class assigned to each instance is obtained. That is, it includes the model's construction (if necessary), any other additional operations to adjust the classifier, and classification of the test partition.

4.6. Statistical procedures

Once an experimental study has been carried out and its main results have been gathered, researchers can start to analyze the performance of the methods considered. For the sake of correctness, these kinds of analysis often require the use of statistical procedures to provide a proper statistical support.

When using this framework, we recommend the consideration of the use of nonparametric statistical tests [88]. Their use is preferred over parametric ones when the initial conditions that guarantee the reliability of the parametric tests (independence, normality and homocedasticity) may not be satisfied, which is a common issue in many machine learning experimental set-ups [33,32].

Several nonparametric procedures are suitable for the aforementioned cases. Specifically, for multiple comparisons involving several procedures, we will consider the use of the Friedman test, together with a *post hoc* procedure for analyzing families of interrelated hypotheses, namely the Shaffer *post hoc* procedure. This set of statistical methods will allow us to contrast and confirm the results obtained in the experimental studies carried out [34].

The Friedman test [30] can be used to test the hypothesis of equality of medians between the results of the algorithms. It works by converting the original results to ranks as follows:

1. Gather observed results for each pair algorithm/data set.
2. For each data set i, rank values from 1 (best algorithm) to k (worst algorithm). Denote these ranks as r_i^j $(1 \leqslant j \leqslant k)$.
3. For each algorithm j, average the ranks obtained in all data sets to obtain the final rank $R_j = \frac{1}{n}\sum_i r_i^j$.

Ranks are then used to compute the test statistic F

$$F_f = \frac{12n}{k(k+1)}\left[\Sigma_j R_j^2 - \frac{k(k+1)^2}{4}\right] \tag{3}$$

which determines whether the hypothesis of equality will be rejected or not.

If the Friedman test's hypothesis of equality is rejected (that is, a low p-value is obtained through the adjustment of F), then it is assumed that there are significant differences among the different algorithms of the experiment.

These differences can then be assessed by using a *post hoc* method. In our case, the Shaffer procedure [87] allows pairwise comparisons to be safely defined, computing a second p-value related to the equality hypothesis between two specific algorithms.

In summary, the joint use of Friedman's and Shaffer's procedures will provide a first p-value (Friedman's test output) determining the degree at which significant differences are present among the algorithms of the experiment, and a set of p-values (Shaffer's procedure output, one per each pair of algorithms) which represent which pairs of algorithms have a significantly different performance. Naturally, the p-value associated with the Friedman test can be expected to be much lower than Shaffer's ones, since the existence of general differences is a necessary condition before finding significant differences between a specific pair of algorithms.

More information about these tests and other statistical procedures specifically designed for use in the field of machine learning can be found at the SCI2S thematic public website on Statistical Inference in Computational Intelligence and Data Mining (http://sci2s.ugr.es/sicidm).

5. A case study: experimental comparison between fuzzy and crisp nearest neighbor classifiers

In this section a case study analyzing the behavior of fuzzy nearest neighbor classifiers is conducted, based on the experimental framework already described. The experimental study is divided into three stages:

- A first stage (Section 5.1) testing the performance of the fuzzy nearest neighbor classifiers over the full collection of data sets included in the framework.
- A second stage (Section 5.2) featuring a comparison of the best performing fuzzy nearest neighbor classifiers with a selection of state of the art crisp nearest neighbor classifiers.
- A third stage (Section 5.3) analyzing the relative improvement in performance of the best fuzzy nearest neighbor classifiers with respect to k-NN, from the point of view of the number of instances, attributes and classes of the data sets.

The purpose of this study is threefold: Firstly, it provides some insights into the current state of fuzzy nearest neighbor classification, when standard supervised problems are considered. Secondly, the behavior of the best performing methods is characterized in a general nearest neighbor classification scenario. And finally, it serves as an example of the experimental framework proposed in this work, demonstrating how to make the most of its main features.

An extended version of the definitions and results obtained in this experimental study is publicly available at http://sci2s.ugr.es/fuzzyKNN/study.php. This version includes training accuracy, test accuracy (Fixed K and Best), training kappa, test kappa (Fixed K and Best) and running time detailed results per each algorithm and data set.

5.1. First stage: comparison of fuzzy nearest neighbor classifiers

In this first stage of the study, we have considered all the fuzzy nearest neighbors classifiers implemented in the library (excepting JFKNN, as it is unable to tackle the largest data sets in a reasonable running time). Average accuracy and kappa results have been collected in two different ways:

- Firstly, a fixed value of k has been selected for each classifier, according to the average accuracy/kappa obtained with each different set-up ($k \in \{3,5,7,9\}$, as noted in Section 4.4). The results obtained using this fixed value of k (the best among the four possibilities) have been termed **Accuracy/Kappa (Fixed k)** results.
- Secondly, the best average accuracy/kappa value per data set has been chosen (considering $k \in \{3,5,7,9\}$ again). The average results obtained using the best value of k for each data set have been termed **Accuracy/Kappa Best)** results.

Table 4 shows the results obtained, sorted from best performing (lowest value in running time column, greatest value otherwise) to worst. For each algorithm and performance measure (accuracy and kappa considering Fixed k and Best k values, and running time) an average value is reported. For fixed k performance measures, the value of k chosen is also shown. A $*$ symbol is used for those methods which do not require the value of k to be fixed. In this case, the results refer to their best configuration (as noted in Section 4.4) and their best configuration per data set.

Note that the results have been obtained through gathering every single result obtained by each algorithm, data set and cross validation partition. For the sake of simplicity, these results have been averaged to obtain a single value per algorithm and data set (as is usually recommended in supervised classification experimental studies).

These results can be contrasted by using the Friedman statistical test. After analyzing the average results obtained regarding accuracy (with a fixed k value), the test reports a p-value of 1.38×10^{-10}, which means that significant differences are found among the algorithms. Using the Shaffer post hoc procedure, 66 differences (out of 153 pairwise comparisons) are found to be significant at a $\alpha = 0.1$ level. Table 5 summarizes the results of both tests, including for each algorithm the rank obtained in the Friedman test and the number of methods for which it is statistically better (+) or equal or better (±) at two different significance levels ($\alpha = 0.1$ and $\alpha = 0.01$, considering the adjusted p-values computed by the Shaffer test).

The results shown in both tables can be analyzed as follows:

- Considering accuracy results with a fixed value of k, the best algorithms are IT2FKNN, GAFuzzyKNN and FuzzyKNN. If the algorithms are compared considering their respective categories in the taxonomy, the best performing algorithms are FuzzyKNN (Fuzzy Sets), IT2FKNN (Type-2 Fuzzy Sets), D-SKNN (Possibilistic methods), IF-KNN (Intuitionistic Fuzzy Sets), FRNN-FRS (Fuzzy Rough Sets) and FENN (Preprocessing Methods via Data Reduction). It is also noticeable that low values for the k parameter (3 and 5) produce better results for most of the methods, excepting IT2FKNN and PFKNN.
- Considering accuracy results with the best value of k, most of the former conclusions hold. However, in this case GAFuzzyKNN achieves the best accuracy result and the differences between D-SKNN and the top 3 algorithms are lower. In general, all methods benefit if the best value of k is chosen for each particular data set, although D-SKNN, IFSKNN and IFV-NP are the methods which obtain a greater benefit (more than 0.01 additional accuracy, on average).
- Considering the kappa performance measure with a fixed value of k, the best algorithms remain IT2FKNN, GAFuzzyKNN and FuzzyKNN. However, in this case GAFuzzyKNN is highlighted as the best method of the Fuzzy Sets family. Other noticeable differences are the relative improvement achieved by FRNN-FRS and FRNN-VQRS, and the performance drop suffered by FENN and PFKNN. Regarding the value of the k parameter, in this case medium values (5 and 7) are generally preferred by the best performing algorithms, with the exception of IF-KNN and FRNN-FRS.
- Considering the best value of k in the analysis with the kappa measure, D-SKNN can also be considered to be the best algorithm (together with IT2FKNN, GAFuzzyKNN and FuzzyKNN). FENN achieves a better relative result and the relative performance of FRNN-VQRS is diminished. Again, all methods benefit if the best value of k is chosen for each particular data set, but the greater improvement is obtained by D-SKNN (more than 0.02 additional kappa, on average).
- Finally, when running time is considered, the most noticeable result is the high computational cost of the GAFuzzyKNN and PFKNN methods (due to their wrapped based nature). The rest of the methods are relatively cheap, computationally speaking, and PosIBL, FRNN-VQRS and FRNN-FRS, and D-SKNN obtain better results in this category. FuzzyNPC is, by far, the most efficient method. However, this contrasts with its poor results in all precision measures.

Table 4
Summary results obtained in the first stage: fuzzy nearest neighbor classifiers.

Accuracy (Fixed k)		k	Accuracy (Best)		Kappa (Fixed k)		k	Kappa (Best)		Running time (s)	
GAFuzzyKNN	0.8130	5	GAFuzzyKNN	0.8204	GAFuzzyKNN	0.6415	5	GAFuzzyKNN	0.6558	FuzzyNPC	0.0409
IT2FKNN	0.8111	7	FuzzyKNN	0.8190	IT2FKNN	0.6354	7	FuzzyKNN	0.6524	PosIBL	2.7363
FuzzyKNN	0.8110	5	IT2FKNN	0.8181	FuzzyKNN	0.6366	7	IT2FKNN	0.6484	FRNN-VQRS	2.9070
D-SKNN	0.7985	5	D-SKNN	0.8136	D-SKNN	0.6167	5	D-SKNN	0.6468	FRNN-FRS	3.0145
IF-KNN	0.7972	3	IF-KNN	0.8062	IF-KNN	0.6157	3	IF-KNN	0.6321	D-SKNN	3.0955
FENN	0.7926	5	FENN	0.8009	FRNN-FRS	0.6130	3	FENN	0.6150	FCMKNN	4.0568
PosIBL	0.7883	$*$	PFKNN	0.7961	PosIBL	0.6071	$*$	FRNN-FRS	0.6138	VWFuzzyKNN	5.6256
PFKNN	0.7877	9	PosIBL	0.7913	FRNN-VQRS	0.6061	5	PosIBL	0.6134	IFSKNN	6.4927
FRNN-FRS	0.7875	3	FRNN-FRS	0.7880	FENN	0.5993	5	PFKNN	0.6130	FuzzyKNN	6.5322
FRNN-VQRS	0.7799	5	VWFuzzyKNN	0.7869	PFKNN	0.5992	7	FRNN-VQRS	0.6104	CFKNN	6.7276
VWFuzzyKNN	0.7775	3	FRNN-VQRS	0.7825	VWFuzzyKNN	0.5793	3	VWFuzzyKNN	0.5936	FENN	6.9731
FRKNNA	0.7640	3	FRKNNA	0.7738	IFSKNN	0.5705	3	IFSKNN	0.5890	FRKNNA	7.2246
IFSKNN	0.7585	5	IFSKNN	0.7713	FRKNNA	0.5612	3	FRKNNA	0.5779	IF-KNN	7.9749
FRNN	0.7408	$*$	FRNN	0.7408	FuzzyNPC	0.5079	$*$	FuzzyNPC	0.5079	IFV-NP	11.1111
FuzzyNPC	0.6975	$*$	FuzzyNPC	0.6975	CFKNN	0.4925	3	CFKNN	0.5000	IT2FKNN	13.1984
CFKNN	0.6885	3	CFKNN	0.6931	FRNN	0.4403	$*$	FCMKNN	0.4497	FRNN	28.5193
FCMKNN	0.6397	5	FCMKNN	0.6469	FCMKNN	0.4390	3	FRNN	0.4403	PFKNN	725.8243
IFV-NP	0.6085	$*$	IFV-NP	0.6337	IFV-NP	0.4153	$*$	IFV-NP	0.4299	GAFuzzyKNN	1275.4415

Table 5
Summary results of Friedman and Shaffer tests for accuracy (fixed *k*, Stage 1).

		$\alpha = 0.1$		$\alpha = 0.01$	
Algorithm	Rank	+	±	+	±
IT2FKNN	4.9659	10	18	9	18
FuzzyKNN	5.3409	10	18	8	18
GAFuzzyKNN	5.3523	10	18	8	18
D-SKNN	6.6818	6	18	5	18
IF-KNN	7.0909	6	18	5	18
FENN	7.2614	5	18	5	18
PFKNN	7.9318	5	18	4	18
PosIBL	8.6023	4	18	3	18
FRNN-FRS	9.2386	3	15	2	18
VWFuzzyKNN	9.7386	2	15	2	17
FRNN-VQRS	9.9091	2	15	2	15
IFSKNN	10.1591	2	15	1	15
FRNN	10.8977	1	13	0	15
FuzzyNPC	12.1250	0	12	0	12
FRKNNA	12.8409	0	11	0	11
CFKNN	13.2841	0	9	0	10
FCMKNN	14.5114	0	6	0	7
IFV-NP	15.0682	0	5	0	6

The analysis performed by the Friedman and Shaffer tests considering accuracy confirms these results: The ranks obtained by the Friedman test are very similar to the relative position of each algorithm regarding accuracy with fixed *k*. Regarding the pairwise comparisons (those analyzed by the Shaffer test) IT2FKNN, GAFuzzyKNN and FuzzyKNN are the best algorithms of the study, showing significant differences with 10 out of the rest of the methods at a $\alpha = 0.1$ significance level (8–9 at a $\alpha = 0.01$ significance level). Moreover, all of the methods highlighted as the best performing of each category of the taxonomy are equal to or better than the rest (± = 18) except FRNN-FRS (for which ± = 15).

The results obtained enable us to make several suggestions and recommendations regarding the use of these fuzzy nearest neighbors classifiers, depending on the performance desired for a specific task:

- If very high accuracy is required, then GAFuzzyKNN, IT2FKNN or FuzzyKNN should be selected given their outstanding overall performance by this measure. However, the high computational cost of GAFuzzyKNN should also be considered if this technique is chosen. Other suitable options for high accuracy without a large running time are IF-KNN and D-SKNN. FENN could also be chosen as an accurate method with the added feature of the removal of noisy instances from the training set, which should also help in reducing the running time in the final classification phase.
- There are very few differences if kappa is considered instead of accuracy. FRNN-FRS shows a small improvement in its results, which suggests that it keeps a better balance (when compared with other methods with similar performances) over the classes of the problems, without bias towards the majority classes. Apart from that, the lack of differences between accuracy and kappa suggest that the best performing algorithms are not biased toward obtaining a good precision in the majority classes of the problems, thus balancing their efforts over all the classes of the domains considered.
- D-SKNN should be the technique to select if a good performance is required without consuming too many computational resources. It is one of the fastest methods analyzed in the study and is not outperformed by any other fuzzy nearest neighbor method, thus becoming a fast and reliable choice in these kinds of situations. A second recommendation would be FRNN-FRS, which is slightly faster than D-SKNN while maintaining good precision rates.

5.2. Second stage: comparison with crisp nearest neighbor approaches

In the second stage of the study, a comparison including the best performing fuzzy nearest neighbor classifiers and several crisp nearest neighbor classifiers will be carried out. The 7 fuzzy nearest neighbor classifiers selected are the best performing methods of each category of the taxonomy; that is, FuzzyKNN and GAFuzzyKNN (Fuzzy Sets), IT2FKNN (Type-2 Fuzzy Sets), D-SKNN (Possibilistic methods), IF-KNN (Intuitionistic Fuzzy Sets), FRNN-FRS (Fuzzy Rough Sets) and FENN (Preprocessing Methods via Data Reduction). As crisp nearest neighbor classifiers, the 7 methods described in Section 4.3 are considered.

Table 6 shows the results obtained in this second stage, following the same experimental conditions as in the first stage.

These results are also contrasted by using the Friedman statistical test. After analyzing the average results obtained regarding accuracy (with a fixed *k* value), the test reports a *p*-value of 1.17×10^{-6}, which means that significant differences are found among the algorithms. Using the Shaffer *post hoc* procedure, 10 differences (out of 91 pairwise comparisons) are found to be significant at a $\alpha = 0.1$ level. Table 7 summarizes the results of both tests, including the rank obtained by each algorithm in the Friedman test and the number of methods for which it is statistically better (+) or equal or better (±) at two different significance levels ($\alpha = 0.1$ and $\alpha = 0.01$, considering the adjusted *p*-values computed by the Shaffer test).

Table 6
Summary results obtained in the second stage: fuzzy and crisp nearest neighbor classifiers.

Accuracy (Fixed k)		k	Accuracy (Best)		Kappa (Fixed k)		k	Kappa (Best)		Running time (s)	
GAFuzzyKNN	0.8130	5	GAFuzzyKNN	0.8204	GAFuzzyKNN	0.6415	5	GAFuzzyKNN	0.6558	FRNN-FRS	3.0145
IT2FKNN	0.8111	7	FuzzyKNN	0.8190	FuzzyKNN	0.6366	7	FuzzyKNN	0.6524	D-SKNN	3.0955
FuzzyKNN	0.8110	5	IT2FKNN	0.8181	IT2FKNN	0.6354	7	IT2FKNN	0.6484	KNN	3.3452
D-SKNN	0.7985	5	D-SKNN	0.8136	D-SKNN	0.6167	5	D-SKNN	0.6468	NSC	4.8859
IF-KNN	0.7972	3	KSNN	0.8098	KSNN	0.6160	3	NSC	0.6379	ENN	5.0116
KSNN	0.7970	5	IF-KNN	0.8062	IF-KNN	0.6157	3	KSNN	0.6359	KNNAdaptive	6.1195
FENN	0.7926	5	NSC	0.8020	FRNN-FRS	0.6130	3	IF-KNN	0.6321	KSNN	6.2721
IDIBL	0.7902	*	FENN	0.8009	KNN	0.6028	7	FENN	0.6150	FuzzyKNN	6.5322
FRNN-FRS	0.7875	3	KNN	0.7933	FENN	0.5993	5	KNN	0.6143	FENN	6.9731
KNNAdaptive	0.7856	3	KNNAdaptive	0.7927	PW	0.5955	*	FRNN-FRS	0.6138	IF-KNN	7.9749
KNN	0.7815	7	IDIBL	0.7902	NSC	0.5814	*	KNNAdaptive	0.6131	IT2FKNN	13.1984
NSC	0.7801	*	ENN	0.7901	IDIBL	0.5807	*	PW	0.6042	PW	22.7235
PW	0.7793	*	FRNN-FRS	0.7880	ENN	0.5740	3	IDIBL	0.5946	IDIBL	409.3493
ENN	0.7784	5	PW	0.7828	KNNAdaptive	0.5697	3	ENN	0.5936	GAFuzzyKNN	1275.4415

Table 7
Summary results of Friedman and Shaffer tests for accuracy (fixed k, Stage 2).

		$\alpha = 0.1$		$\alpha = 0.01$	
Algorithm	Rank	+	±	+	±
IT2FKNN	5.5795	4	18	2	18
GAFuzzyKNN	5.8295	3	18	1	18
FuzzyKNN	5.8864	3	18	1	18
KSNN	6.5455	0	18	0	18
KNNAdaptive	6.5682	0	18	0	18
D-SKNN	7.3977	0	18	0	18
IF-KNN	7.4318	0	18	0	18
KNN	7.6591	0	18	0	18
FENN	7.8409	0	18	0	18
IDIBL	8.3295	0	18	0	18
PW	8.5455	0	17	0	18
ENN	8.9659	0	15	0	18
FRNN-FRS	9.1023	0	15	0	17
NSC	9.3182	0	15	0	15

The results shown in both tables can be analyzed as follows:

- If accuracy with a fixed value of k is considered, the five best positions are achieved by fuzzy nearest neighbor classifiers (GAFuzzyKNN, IT2FKNN, FuzzyKNN, D-SKNN and IF-KNN). None of the 7 fuzzy nearest neighbor classifiers included shows a performance lower than the original k-NN rule. It is also interesting to note the improvement achieved by FuzzyKNN and FENN over their direct crisp counterparts, KNN and ENN (an improvement of 0.0295 and 0.0142, respectively).
- The former results hold, in general, if the best value of k is considered individually. Also, in this case, the crisp methods KSNN and NSC shows a performance comparable with some of the best fuzzy nearest neighbor classifiers. FRNN-FRS is the only fuzzy nearest neighbor classifiers whose performance drops below KNN under these conditions.
- The results obtained using the kappa performance measure with a fixed value of k continue to highlight GAFuzzyKNN, IT2FKNN and FuzzyKNN as the better algorithms. KSNN and FRNN-FRS show an improvement, being comparable to D-SKNN and IF-KNN in this category. In this case, only FENN's kappa falls below KNN's.
- Considering the best value of k with the kappa measure, the differences among methods are more tight. Most of the former results are the same, with the most noticeable differences being the improvement achieved by NSC and the relative drop in the position of FRNN-FRS (which makes almost no improvement by allowing the best value of k to be set in each data set).
- Finally, there are not many differences between crisp and fuzzy methods with respect to running time. Both families have very fast methods (FRNN-FRS, D-SKNN, KNN, NSC and ENN) and slower ones (IDIBL and GAFuzzyKNN), which shows that there is not an additional computational cost when fuzzy mechanisms are introduced to improve the nearest neighbor rule, in comparison with crisp based mechanisms.

The analysis performed by the Friedman and Shaffer tests considering accuracy confirms the superiority of GAFuzzyKNN, IT2FKNN and FuzzyKNN in terms of accuracy, as they are the only methods able to improve statistically some of the rest of the classifiers of the comparison, both at a $\alpha = 0.1$ and at a $\alpha = 0.01$ significance level. PW, ENN, FRNN-FRS and NSC are the methods improved by the former ones, in this sense.

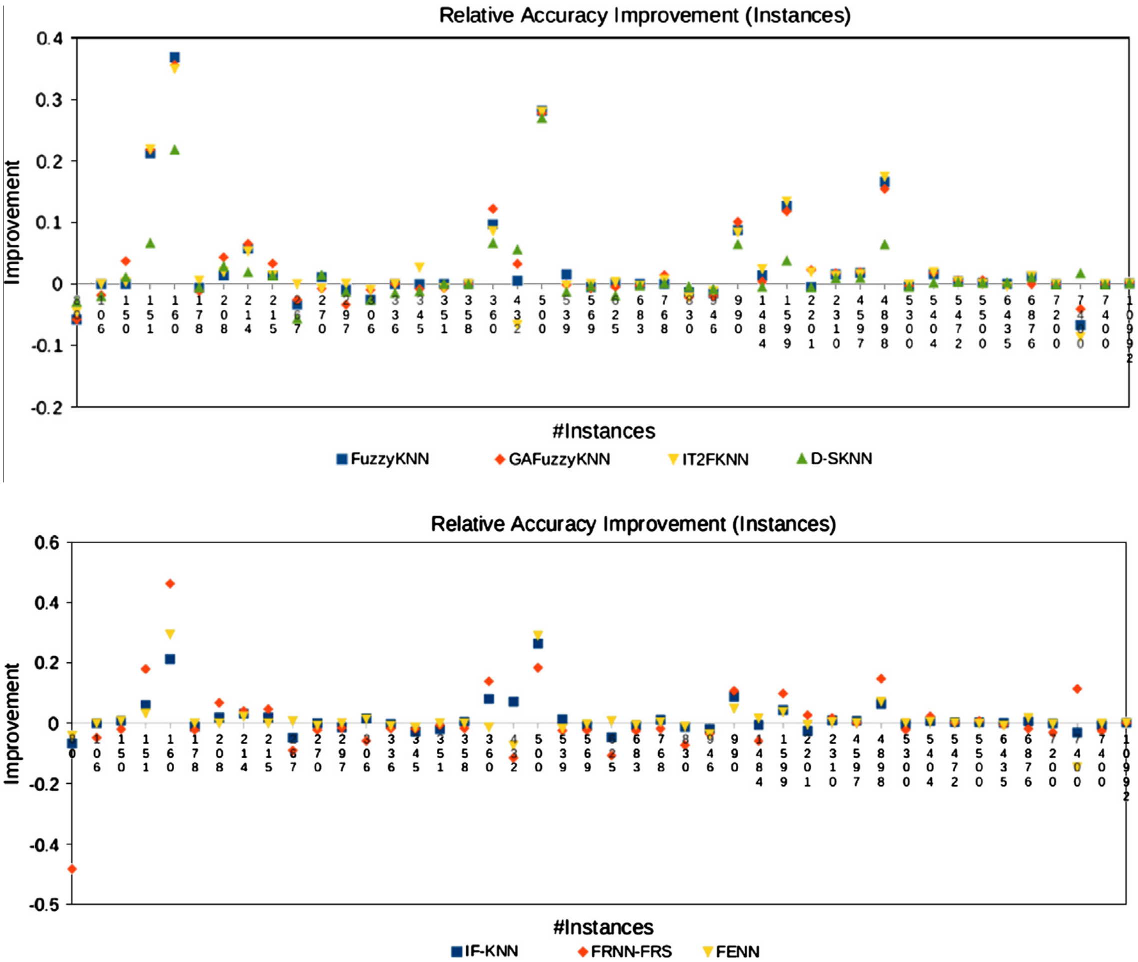

Fig. 2. Relative accuracy improvement between *k*-NN and the 7 best fuzzy nearest neighbor algorithms of the study. Data sets are ordered with respect to the number of instances.

In general, this second study has shown that fuzzy nearest neighbor classifiers can demonstrate a positive performance if considered within the state of the art in nearest neighbor classification: Several methods offer better precision (accuracy and kappa rates) and they are neither particularly slower or faster than the crisp approaches. Hence, they are a suitable option for standard supervised learning tasks, in which high accuracy at a relatively lower computational cost is required.

5.3. Third stage: analysis of performance with respect to the number of instances, attributes and classes

The last stage of the study is devoted to analyzing the behavior of the fuzzy nearest neighbor classifiers, as the number of instances, attributes and classes changes.

Specifically, we have selected the 7 best performing fuzzy nearest nearest neighbor methods (the same that were chosen in Section 5.2) and we have considered their accuracy results (with a fixed *k* value). Then, we have computed their relative accuracy improvement with respect to *k*-NN, for each of the 44 data sets included in the experimental framework.

Fig. 2 shows a graphic representation of the accuracy of the methods as the number of instances rises. Results have been sorted according to the number of instances of each data set (shown on the *X*-axis of the figure). Relative accuracy improvement over *k*-NN is represented on the *Y*-axis. Also, Figs. 3 and 4 show similar graphics where results have been ordered with respect to attibutes and classes, respectively.

The main conclusions that can be drawn from these figures are as follows:

- Instances: The differences between the fuzzy nearest neighbor methods and *k*-NN tend to narrow down as the number of instances increases, in general. This suggests that fuzzy nearest neighbor algorithms can be useful in those cases in which the amount of data available is not enough to represent the classification problem properly. As the available data increases, the impact of this property appears to dimish.

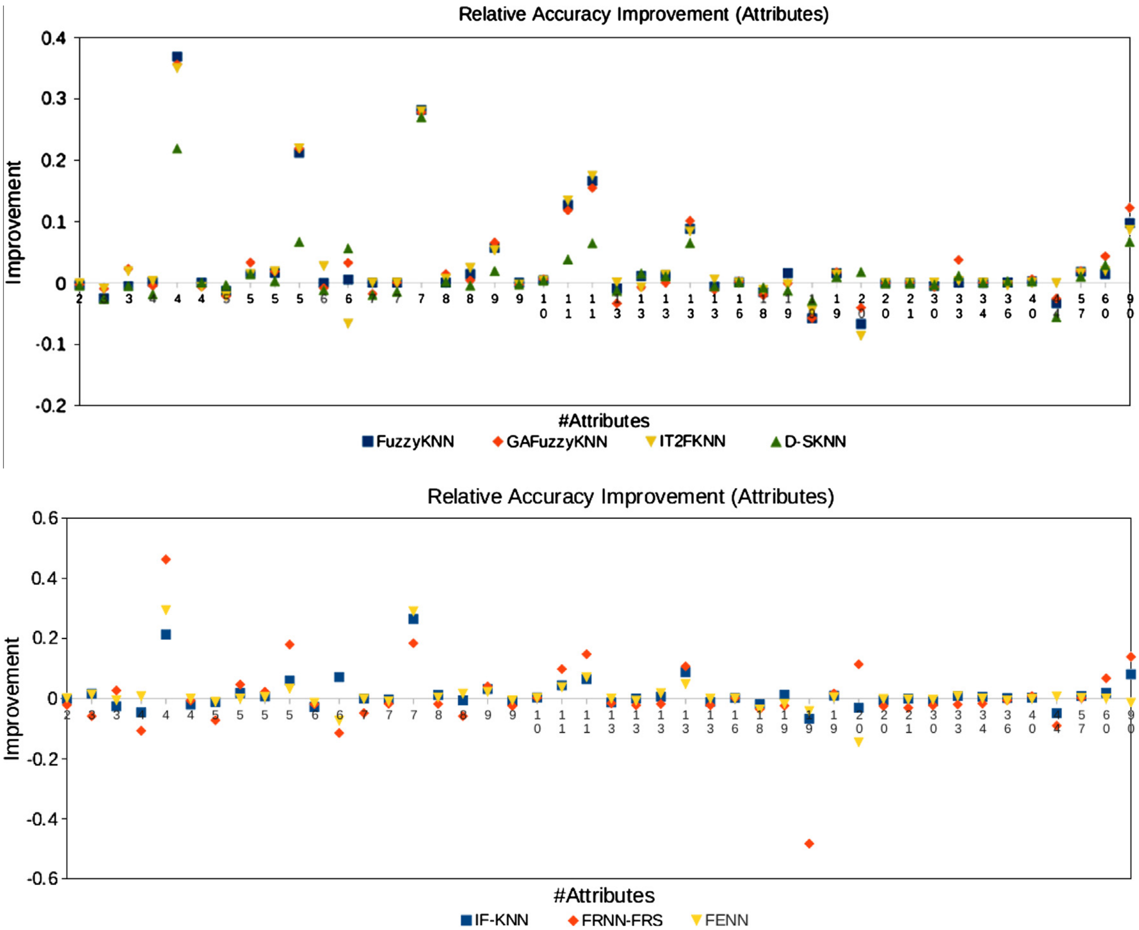

Fig. 3. Relative accuracy improvement between *k*-NN and the 7 best fuzzy nearest neighbor algorithms of the study. Data sets are ordered with respect to the number of attributes.

- Features: In this case, it is possible to draw similar conclusions to the above, although it is important to note that there is a substantial improvement in the high dimensional data sets of the study (those with more than 50 attributes). This last result suggests that these methods (except FENN) may be useful when trying to enhance the behavior of the NN rule in high dimensional problems.
- Classes: When the number of classes is considered, it can be shown that the relative differences rise as this number grows. Fuzzy nearest neighbor methods perform similarly to *k*-NN in two-class problems, but differences increase in muti-class problems. The most clear differences are obtained with 10 or more classes. This result is the consequence of one of the general strengths of fuzzy nearest neighbor algorithms: The capability of managing different degrees of memberships to the classes per each instance. This capability allows the classifiers to refine their output in those problems whose difficulty increases due to the presence of many classes.

To summarize, fuzzy nearest neighbor methods can be very competitive when compared with the *k*-NN classifier, showing a similar or better performance with respect to accuracy. The differences found are greater if the available data is not enough for *k*-NN to fully characterize the domain of the problem, in high dimensional problems and, particularly, when problems with multiple classes are considered.

6. Future prospects

The experimental study performed has shown the general capabilities of fuzzy nearest neighbor classifiers. The methods have been compared, and have also been tested against a set of general nearest neighbor classifiers, revealing that they achieve a promising performance in general supervised classification problems.

The conclusions drawn throughout the survey can be used to suggest some unaddressed challenges which could be very valuable to the further development of the field:

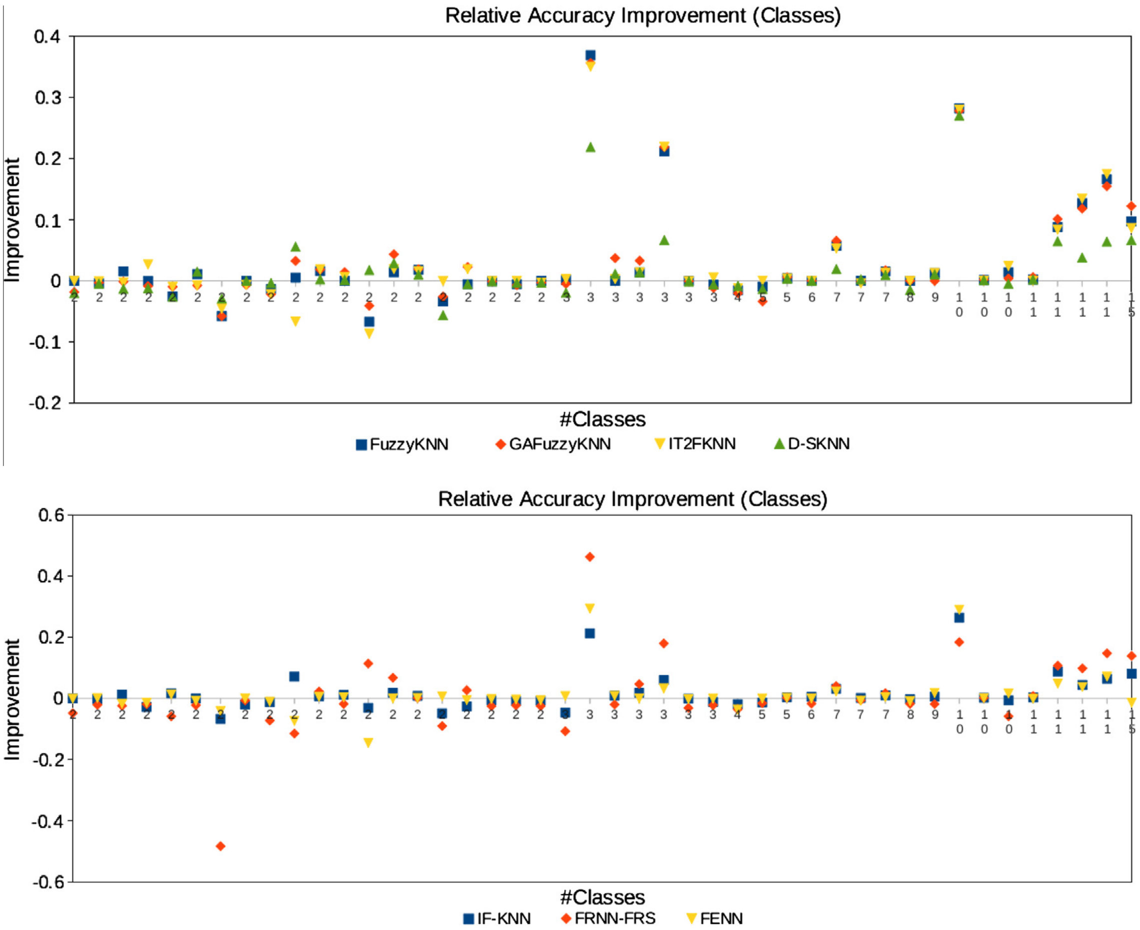

Fig. 4. Relative accuracy improvement between *k*-NN and the 7 best fuzzy nearest neighbor algorithms of the study. Data sets are ordered with respect to the number of classes.

- Most of the techniques reviewed are not able to dealt properly with nominal (categorical) attributes and missing values, or do not directly describe a method for managing them. Although some solutions can be incorporated from the classical classification field (such as advanced similarity measures for tackling nominal attributes [97] or imputation techniques for handling missing data [72]), there remains the necessity of a specific solution which will enable fuzzy nearest neighbor classifiers to handle these kinds of data with ease.
- A key aspect in the performance of the fuzzy nearest neighbor algorithms is the way in which the membership values to the classes are computed. It is true that there is a wide variety of ways in which these values may be represented. However, most of them are based on the concept of locality (that is, membership is assigned in accordance with the nearest instances in the training data). Other schemes of analysis, based on different concepts such as the global characteristics of the data, could be incorporated to develop new membership assignation schemes, likely to further improve the generalization capabilities of the algorithms. The development of a new class of preprocessing techniques could also be helpful here, if they are to be applied as a way of refining an initial configuration of memberships for a data set.
- Using the theoretical developments shown in [102] as a starting point, new voting schemes could be designed (probably in an automatic way), far from the traditional majority rules or the search for a best single instance. The definition of *ad hoc* voting rules, specific to the current problem tackled by the classifier, would enable a specialized treatment of the intrinsic characteristics of the data. These rules would allow the classifier to be fitted to the problem addressed, further enhancing its classification performance.

7. Conclusions

In this work we have presented a survey of fuzzy nearest neighbor classifiers. The application of FST and some of its extensions to the development of enhanced nearest neighbor algorithms have been reviewed, from the very first proposals to the most recent approaches. Several discriminating traits of the techniques has been described as the building blocks of a multi-level taxonomy, devised to accommodate present and future proposals with ease.

An experimental framework is provided, incorporating implementations of the most relevant algorithms in the state of the art. A case study is then conducted, testing the performance of the fuzzy nearest neighbor classifiers. The experiment also includes a further comparison with several state of the art crisp nearest neighbor classifiers. The conclusions of the study reveal which are the most desirable fuzzy nearest neighbor classifiers according to several performance measures, and note the competitiveness of these techniques in comparison to the classical nearest neighbor based approaches.

As a final remark, we would like to note that there is a dedicated website providing all the complementary material to the paper (algorithms and data sets of the experimental framework, and extended results and statistical analysis conducted in the case study). These contents can be retrieved at http://sci2s.ugr.es/fuzzyKNN.

Acknowledgements

This work was partially supported by the Spanish Ministry of Science and Technology under project TIN2011-28488 and the Andalusian Research Plans P11-TIC-7765 and P10-TIC-6858.

References

[1] D.W. Aha, D. Kibler, M.K. Albert, Instance-based learning algorithms, Machine Learning 6 (1991) 37–66.
[2] J. Alcalá-Fdez, A. Fernández, 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 (2011).
[3] J. Alcalá-Fdez, L. Sánchez, S. García, M.J. del Jesus, S. Ventura, J.M. Garrell, J. Otero, C. Romero, J. Bacardit, V.M. Rivas, J.C. Fernández, F. Herrera, KEEL: a software tool to assess evolutionary algorithms for data mining problems, Soft Computing 13 (2008) 307–318.
[4] E. Alpaydin, Introduction to Machine Learning, second ed., The MIT Press, 2010.
[5] A. Andoni, P. Indyk, Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions, Communications of the ACM 51 (2008).
[6] M. Arif, M.U. Akram, F.A. Afsar, F.A.A. Minhas, Pruned fuzzy *k*-nearest neighbor classifier for beat classification, Journal of Biomedical Science and Engineering 3 (2010).
[7] K. Atanassov, Intuitionistic fuzzy sets, Fuzzy Sets and Systems 20 (1986) 87–96.
[8] J. Bedzek, P. Castelaz, Prototype classification and feature selection with fuzzy sets, IEEE Transactions on Systems, Man, and Cybernetics 7 (1977) 87–92.
[9] J. Bedzek, S.K. Chuah, Generalized *k*-nearest neighbor rules, Fuzzy Sets and Systems 18 (1986) 237–256.
[10] A. Ben-David, A lot of randomness is hiding in accuracy, Engineering Applications of Artificial Intelligence 20 (2007) 875–885.
[11] M. Béreau, B. Dubuisson, A fuzzy extended *k*-nearest neighbors rule, Fuzzy Sets and Systems 44 (1991) 17–32.
[12] H. Bian, L. Mazlack, Fuzzy-rough nearest neighbor classification approach, in: Proceedings of the 22th International Conference of the North American Fuzzy Information Processing Society (NAFIPS'03), Chicago, Illinois, USA, July 24–26, pp. 500–505.
[13] D. Cabello, S. Barro, J. Salceda, R. Ruiz, J. Mira, Fuzzy *k*-nearest neighbor classifiers for ventricular arrhytmia detection, International Journal of Biomedical Computing 27 (1991) 77–93.
[14] O. Castillo, P. Melin, E. Ramírez, J. Soria, Hybrid intelligent system for cardiac arrhythmia classification with fuzzy *k*-nearest neighbors and neural networks combined with a fuzzy system, Expert Systems with Applications 39 (2012) 2947–2955.
[15] H.L. Chen, B. Yang, G. Wang, J. Liu, X. Xu, S. Wang, D. Liu, A novel bankruptcy prediction model based on an adaptive fuzzy *k*-nearest neighbor method, Knowledge Based Systems 24 (2011) 1348–1359.
[16] Y. Chen, E.K. Garcia, M.R. Gupta, A. Rahimi, L. Cazzanti, Similarity-based classification: concepts and algorithms, Journal of Machine Learning Research 10 (2009) 747–776.
[17] M.A. Chikh, M. Saidi, N. Settouti, Diagnosis of diabetes diseases using an artificial immune recognition system2 (AIRS2) with fuzzy *k*-nearest neighbor, Journal of medical systems 36 (2012) 2721–2729.
[18] T.W. Chua, K. Leman, N.T. Pham, Human action recognition via sum-rule fusion of fuzzy *k*-nearest neighbor classifiers, in: IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2011), Taipei, Taiwan, June 27–30, pp. 484–489.
[19] T.W. Chua, W.W. Tan, A new fuzzy rule-based initialization method for *k*-nearest neighbor classifier, in: Proceedings of the IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'09), Jeju island, Korea, August 20–24, pp. 415–420.
[20] F. Chung-Hoon, C. Hwang, An interval type-2 fuzzy *k*-nearest neighbor, in: Proceedings of the 12th IEEE International Conference on Fuzzy Systems (IEEE FUZZY'03), St. Louis, Missouri, USA, May 25–28, pp. 802–807.
[21] J. Cohen, A coefficient of agreement for nominal scales, Educational and Psychological Measurement 20 (1960) 37–46.
[22] C. Cornelis, R. Jensen, G. Hurtado, D. Slezak, Attribute selection with fuzzy decision reducts, Information Sciences 180 (2010) 209–224.
[23] T.M. Cover, P.E. Hart, Nearest neighbor pattern classification, IEEE Transactions on Information Theory 13 (1967) 21–27.
[24] M. De Cock, C. Cornelis, E.E. Kerre, Fuzzy rough sets: the forgotten step, IEEE Transactions on Fuzzy Systems 15 (2007) 121–130.
[25] T. Denoeux, A *k*-nearest neighbor classification rule based on Dempster–Shafer theory, IEEE Transactions on Systems, Man, and Cybernetics 25 (1995) 804–813.
[26] J. Derrac, C. Cornelis, S. García, F. Herrera, Enhancing evolutionary instance selection algorithms by means of fuzzy rough set based feature selection, Information Sciences 186 (2012) 73–92.
[27] S. Destercke, A *k*-nearest neighbours method based on imprecise probabilities, Soft Computing 16 (2012) 833–844.
[28] E. Fix, J. Hodges, Discriminatory analysis, nonparametric discrimination: Consistency properties, Technical Report 4, USAF School of Aviation Medicine, 1951.
[29] A. Frank, A. Asuncion, UCI Machine Learning Repository, 2010.
[30] M. Friedman, The use of ranks to avoid the assumption of normality implicit in the analysis of variance, Journal of the American Statistical Association 32 (1937) 674–701.
[31] S. García, J. Derrac, J.R. Cano, F. Herrera, Prototype selection for nearest neighbor classification: taxonomy and empirical study, IEEE Transactions on Pattern Analysis and Machine Intelligence 34 (2012) 417–435.
[32] 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.
[33] S. García, A. Fernández, 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 (2010) 2044–2064.
[34] S. García, F. Herrera, An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons, Journal of Machine Learning Research 9 (2008) 2677–2694.
[35] N. Gayar, F. Schwenker, G. Palm, A study of the robustness of *k*NN classifiers trained using soft labels, in: Proceedings of the Second IAPR Workshop on Artificial Neural Networks in Pattern Recognition (ANNPR 2006), Ulm, Germany, August 31–September 2, pp. 67–80.
[36] V. Gesu, G. Bosco, Combining one class fuzzy *k*NN's, in: 7th International Workshop on Fuzzy Logic and Applications (WILF 2007), Camogli, Italy, July 7–10, pp. 152–160.

[37] P. Grother, G.T. Candela, J.L. Blue, Fast implementations of nearest-neighbor classifiers, Pattern Recognition 30 (1997) 459–465.
[38] S. Hadjitodorov, An intuitionistic fuzzy sets application to the *k*-NN method, Notes on Intuitionistic Fuzzy Sets 1 (1995) 66–69.
[39] S. Hadjitodorov, An intuitionistic fuzzy version of the nearest prototype classification method, based on a moving-of-pattern procedure, International Journal of General Systems 30 (2000) 155–165.
[40] J.H. Han, Y.K. Kim, A fuzzy *k*-NN algorithm using weights from the variance of membership values, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (IEEE CVPR'99), Ft. Collins, Colorado, USA, June 23–25, pp. 394–399.
[41] P.E. Hart, The condensed nearest neighbor rule, IEEE Transactions on Information Theory 18 (1968) 515–516.
[42] L. Hartert, M.S. Mouchaweh, P. Billaudel, A semi-supervised dynamic version of fuzzy *k*-nearest neighbours to monitor evolving systems, Evolving Systems 1 (2010) 3–15.
[43] T. Hastie, R. Tibshirani, Discriminant adaptive nearest neighbor classification, IEEE Transactions on Pattern Analysis and Machine Intelligence 18 (1996) 607–616.
[44] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning: Data Mining, Inference and Prediction, seciond ed., Springer, New York, Berlin, Heidelberg, 2009.
[45] M. Hayat, A. Khan, Discriminating outer membrane proteins with fuzzy *k*-nearest neighbor algorithms based on the general form of chou's PseAAC, Protein and Peptide Letters 19 (2012) 411–421.
[46] X. Hu, C. Xie, Improving fuzzy *k*-NN by using genetic algorithm, Journal of Computational Information Systems 1 (2005) 203–213.
[47] Z. Huan, N. Yi, H. Nie, Web document classification based on fuzzy *k*-NN algorithm, in: Proceedings of the 2009 International Conference on Computational Intelligence and Security (CIS 2009), Beijing, China, December 11–14, pp. 193–196.
[48] Y. Huang, Y. Li, Prediction of protein subcellular locations using fuzzy *k*-NN method, Bioinformatics 20 (2004) 21–28.
[49] E. Hüllermeier, Possibilistic instance-based learning, Artificial Intelligence 148 (2003) 335–383.
[50] H. Ishibuchi, T. Nakashima, Evolution of fuzzy nearest neighbor neural networks, in: Proceedings of the 1997 IEEE International Conference on Evolutionary Computation (IEEE CEC'97), Indianapolis, USA, April 13–16, pp. 673–678.
[51] M.Z. Jahromi, E. Parvinnia, R. John, A method of learning weighted similarity function to improve the performance of nearest neighbor, Information Sciences 179 (2009) 2964–2973.
[52] R. Jensen, C. Cornelis, Fuzzy-rough instance selection, in: IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2010), Barcelona, Spain, July 18–23, pp. 1776–1782.
[53] R. Jensen, C. Cornelis, Fuzzy-rough nearest neighbour classification, Transactions on Rough Sets 13 (2011) 56–72.
[54] J.Y. Jiang, S.C. Tsai, S.J. Lee, FSKNN: multi-label text categorization based on fuzzy similarity and k nearest neighbors, Expert Systems with Applications 39 (2012) 2813–2821.
[55] K. Josien, T.W. Liao, Integrated use of Fuzzy C-Means and fuzzy *k*NN for gt part family and machine cell formation, International Journal of Production Research 38 (2000) 3513–3536.
[56] A. Jóẃik, A learning scheme for a fuzzy *k*-NN rule, Pattern Recognition Letters 1 (1983) 287–289.
[57] X.M. Kang, X.P. Liu, J.H. Zhai, M.Y. Zhai, Instances selection for NN with fuzzy rough technique, in: Proceedings of the International Conference on Machine Learning and Cybernetics (ICMLC 2011), Guilin, China, July 10–13, pp. 1097–1100.
[58] N. Karnik, J. Mendel, Q. Liang, Type-2 fuzzy logic systems, IEEE Transactions on Fuzzy Systems 7 (1999) 643–658.
[59] J.M. Keller, M.R. Gray, J.A. Givens, A fuzzy *k*-nearest neighbor algorithm, IEEE Transactions on Systems, Man, and Cybernetics 15 (1985) 580–585.
[60] A.K. Ghosh, On optimum choice of k in nearest neighbor classification, Computational Statistics & Data Analysis 50 (2006) 3113–3123.
[61] S. Kim, J. Sim, J. Lee, Fuzzy *k*-nearest neighbor method for protein secondary structure prediction and its parallel implementation, in: Proceedings of the International Conference on Intelligent Computing (ICIC 2006), Kunming, China, August 16–19, pp. 444–463.
[62] V. Kissiov, S. Hadjitodorov, A fuzzy version of the *k*-NN method, Fuzzy Sets and Systems 49 (1992) 323–329.
[63] L.I. Kuncheva, An intuitionistic fuzzy *k*-nearest neighbors rule, Notes on Intuitionistic Fuzzy Sets 1 (1995) 56–60.
[64] M. Kuske, R. Rubio, A.C. Romain, J. Nicolas, S. Marco, Fuzzy *k*-NN applied to moulds detection, Sensors and Actuators B 106 (2004) 52–60.
[65] K. Leszczynski, S. Cosby, R. Bissett, D. Provost, S. Boyko, S. Loose, E. Mvilongo, Application of a fuzzy pattern classifier to decision making in portal verification of radiotherapy, Physics in Medicine and Biology 44 (1999) 253–269.
[66] S. Liang, C. Li, A fast and scalable fuzzy-rough nearest neighbor algorithm, in: Proceedings of the IEEE Global Congress on Intelligent Systems (GCIS'09), Xiamen, China, May 19–21, pp. 311–314.
[67] T. Liao, D. Li, Two manufacturing applications of the fuzzy *k*-NN algorithm, Fuzzy Sets and Systems 92 (1997) 289–303.
[68] T. Liao, D. Li, Y. Li, Detection of welding flaws from radiographic images with fuzzy clustering methods, Fuzzy Sets and Systems 108 (1999) 145–158.
[69] T. Liao, D. Li, Y. Li, Extraction of welds from radiographic images using fuzzy classifiers, Information Sciences 126 (2000) 21–40.
[70] Y.C. Liaw, M.L. Leou, C.M. Wu, Fast exact k nearest neighbors search using an orthogonal search tree, Pattern Recognition 43 (2010) 2351–2358.
[71] F.O.S.S. Lisboa, M. Nicoletti, A. Ramer, A version of the NGE model suitable for fuzzy domains, Journal of Intelligent & Fuzzy Systems 18 (2007) 1–17.
[72] R.J. Little, D. Rubin, Statistical Analysis with Missing Data, second ed., John Wiley and Sons, 2002.
[73] D.Y. Liu, H.L. Chen, B. Yang, X.E. Lv, L.N. Li, J. Liu, Design of an enhanced fuzzy *k*-nearest neighbor classifier based computer aided diagnostic system for thyroid disease, Journal of medical systems 36 (2012) 3243–3254.
[74] J. McNames, A fast nearest-neighbor algorithm based on a principal axis search tree, IEEE Transactions on Pattern Analysis and Machine Intelligence 23 (2001) 964–976.
[75] H.B. Mitchell, P.A. Schaefer, A soft *k*-nearest neighbor voting scheme, International Journal of Intelligent Systems 16 (2001) 459–468.
[76] T. Nakashima, H. Ishibuchi, Learning of fuzzy reference sets in nearest neighbor classification, in: Proceedings of the 18th International Conference of the North American Fuzzy Information Processing Society (NAFIPS'99), New York, USA, June 10–12, pp. 357–360.
[77] R. Nock, M. Sebban, D. Bernard, A simple locally adaptive nearest neighbor rule with application to pollution forecasting, International Journal of Pattern Recognition and Artificial Intelligence 17 (2003) 1369–1382.
[78] R. Ostermark, A fuzzy vector valued *k*NN-algorithm for automatic outlaier detection, Applied Soft Computing 9 (2009) 1263–1272.
[79] R. Paredes, E. Vidal, Learning weighted metrics to minimize nearest-neighbor classification error, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (2006) 1100–1110.
[80] V. Petridis, V.G. Kaburlasos, Finknn: a fuzzy interval number *k*-nearest neighbor classifier for prediction of sugar production from populations of samples, Journal of Machine Learning Research 4 (2004) 17–37.
[81] T.D. Pham, An optimally wieghted fuzzy *k*-NN algorithm, in: Proceedings of the Third International Conference on Advances in Pattern Recognition, Part I (ICAPR 2005), Bath, UK, August 22–25, pp. 239–247.
[82] Y. Qu, C. Shang, Q. Shen, N.M. Parthaláin, W. Wu, Kernel-based fuzzy-rough nearest neighbour classification, in: IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2011), Taipei, Taiwan, June 27–30, pp. 1523–1529.
[83] C. Raptis, C. Siettos, C. Kiranoudis, G. Bafas, Classification of aged wine distillates using fuzzy and neural network systems, Journal of Food Engineering 46 (2000) 267–275.
[84] S. Roh, T. Ahn, W. Pedrycz, The refinement of models with the aid of the fuzzy *k*-nearest neighbors approach, IEEE Transactions on Instrumentation and Measurement 59 (2010) 604–615.
[85] J.L.A. Rosa, N.F.F. Ebecken, Data mining for data classification based on the *k*NN-fuzzy method supported by genetic algorithm, in: Proceedings of the 5th International Conference on High Performance Computing for Computational Science (VECPAR'02), Porto, Portugal, June 26–28, Lecture Notes in Computer Science, vol. 2565, pp. 126–133.
[86] M. Sarkar, Fuzzy-rough nearest neighbor algorithms in classification, Fuzzy Sets and Systems 158 (2012) 2134–2152.
[87] J. Shaffer, Modified sequentially rejective multiple test procedures, Journal of American Statistical Association 81 (1986) 826–831.

[88] D.J. Sheskin, Handbook of Parametric and Nonparametric Statistical Procedures, fifth ed., Chapman & Hall/CRC, 2011.
[89] S. Singh, A single nearest neighbour fuzzy approach for pattern recognition, International Journal of Pattern Recognition and Artificial Intelligence 13 (1998) 49–54.
[90] M. Stone, Cross-validatory choice and assessment of statistical predictions, Journal of the Royal Statistical Society, Series B (Methodological) 36 (1974) 111–147.
[91] I. Triguero, J. Derrac, S. García, F. Herrera, A taxonomy and experimental study on prototype generation for nearest neighbor classification, IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 42 (2012) 86–100.
[92] D.C. Tu, J.H. Zhao, M.H. Liu, J. Shen, F. Yu, Preliminary study on quantification of duck color based on fuzzy k nearest neighbor method, Applied Mechanics and Materials 39 (2010) 210–215.
[93] T.Q. Tung, D. Lee, A method to improve protein subcellular localization prediction by integrating various biological data sources, BMC Bioinformatics 10 (2009) 1–10.
[94] C.J. Veenman, M.J.T. Reinders, The nearest subclass classifier: a compromise between the nearest mean and nearest neighbor classifier, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (2005) 1417–1429.
[95] J. Wang, P. Neskovic, L. Cooper, Improving nearest neighbor rule with a simple adaptive distance measure, Pattern Recognition Letters 28 (2007) 207–213.
[96] D. Wettschereck, D.W. Aha, T. Mohri, A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms, Artificial Intelligence Review 11 (1997) 273–314.
[97] D. Wilson, T. Martinez, Improved heterogeneous distance functions, Journal of Artificial Intelligence Research 6 (1997) 1–34.
[98] D.L. Wilson, Asymptotic properties of nearest neighbor rules using edited data, IEEE Transactions on System, Man and Cybernetics 2 (1972) 408–421.
[99] D.R. Wilson, T.R. Martinez, An integrated instance-based learning algorithm, Computational Intelligence 16 (2000) 1–28.
[100] I.H. Witten, E. Frank, M.A. Hall, Data Mining: Practical Machine Learning Tools and Techniques, third ed., Morgan Kaufmann Series in Data Management Systems, Morgan Kaufmann, 2011.
[101] X. Wu, V. Kumar (Eds.), The Top Ten Algorithms in Data Mining, Data Mining and Knowledge Discovery, Chapman & Hall/CRC, 2009.
[102] R.R. Yager, Using fuzzy methods to model nearest neighbor rules, IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics 32 (2002) 512–525.
[103] M. Yang, C. Chen, On strong consistency of the fuzzy generalized nearest neighbor rule, Fuzzy Sets and Systems 60 (1993) 273–281.
[104] M. Yang, C. Chen, Convergence rate of the fuzzy generalized nearest neighbor rule, Computers & Mathematics with Applications 27 (1994) 1–8.
[105] M. Yang, C. Cheng, On the edited fuzzy *k*-nearest neighbor rule, IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics 28 (1998) 461–466.
[106] S. Yu, S. Backer, P. Scheunders, Genetic feature selection combined with composite fuzzy nearest neighbor classifiers for hyperspectral satellite imagery, Pattern Recognition Letters 23 (2002) 183–190.
[107] L. Zadeh, Fuzzy sets, Information and Control 8 (1965) 338–353.
[108] L. Zadeh, Fuzzy sets as a basis for a theory of possibility, Fuzzy Sets and Systems 1 (1978) 3–28.
[109] J.H. Zhai, N. Li, M.Y. Zhai, The condensed fuzzy *k*-nearest neighbor rule based on sample fuzzy entropy, in: Proceedings of the 2011 International Conference on Machine Learning and Cybernetics (ICMLC'11), Guilin, China, July 10–13, pp. 282–286.
[110] H. Zhu, O. Basir, A *k*-NN associated fuzzy evidential reasoning classifier with adaptive neighbor selection, in: Proceedings of the Third IEEE International Conference on Data Mining (ICDM 2003), Melbourne, Florida, USA, December 19–22, pp. 709–712.