Integrating Instance Selection, Instance Weighting, and Feature Weighting for Nearest Neighbor Classifiers by Coevolutionary Algorithms

Joaquín Derrac, Isaac Triguero, Salvador García, and Francisco Herrera, *Member, IEEE*

***Abstract*—Cooperative coevolution is a successful trend of evolutionary computation which allows us to define partitions of the domain of a given problem, or to integrate several related techniques into one, by the use of evolutionary algorithms. It is possible to apply it to the development of advanced classification methods, which integrate several machine learning techniques into a single proposal. A novel approach integrating instance selection, instance weighting, and feature weighting into the framework of a coevolutionary model is presented in this paper. We compare it with a wide range of evolutionary and nonevolutionary related methods, in order to show the benefits of the employment of coevolution to apply the techniques considered simultaneously. The results obtained, contrasted through nonparametric statistical tests, show that our proposal outperforms other methods in the comparison, thus becoming a suitable tool in the task of enhancing the nearest neighbor classifier.**

***Index Terms*—Cooperative coevolution, feature weighting (FW), instance selection (IS), instance weighting (IW), nearest neighbor rule.**

I. Introduction

CLASSIFICATION is one of the most well-known tasks in machine learning [1]–[4]. Starting from an already processed training set, machine learning methods are able to extract knowledge from the data, which can be used to characterize new samples and classify them into classes already specified by the domain of the problem. Although most of these methods store and represent this knowledge by building a model during their execution, there are some approaches where the construction of this model is not necessary. They are known as lazy learning methods [5].

The most well-known lazy classifier is the k-nearest neighbors (k-NNs) [6], which is one of the most relevant algorithms in data mining [7]. It is a nonparametric classifier which simply uses the entire input data set to establish the classification rule. Thus, the effectiveness of the classification process performed by k-NN relies mainly on the quality of the training data. Also, it is important to note that its main drawback is its relative inefficiency as the size of the problem increases, regarding both the number of examples in the data set and the number of attributes which will be used in the computation of the similarity function (distance) [8].

Many approaches have been proposed to improve the performance of k-NN [9]–[14]. Some of the most effective have been developed for data preparation [15]. Their task is to assess, prepare, and preprocess the initially available data in data mining processes. Their main goal is the improvement of the algorithms in terms of efficiency and efficacy.

One way to prepare a suitable training set is to reduce it. In this sense, data reduction [15] techniques try to obtain a reduced version of the original training set, removing noisy and irrelevant data (which may be harmful to the majority of machine learning methods). Instance selection (IS) [16], which consists of selecting the most appropriate examples (instances) in the training set, will be used in this study.

Another way to improve the performance of k-NN is through the use of weighting schemes. Feature weighting (FW) [11] is a well-known technique which consists of assigning a weight to each feature of the domain of the problem, modifying the way in which distances between examples are computed. The definition of weights associated with the instances [instance weighting (IW)] is also possible. This approach, which has been used to improve the results of some machine learning methods, can also be used to modify the computation of the distance function [17].

Evolutionary algorithms (EAs) [18] are search algorithms that use principles inspired by natural populations to evolve solutions. They have been applied to different data mining problems [19]–[22]. Given that IS, IW, and FW tasks can be defined as combinatorial problems, it is possible to carry them out using EAs [23]. In fact, many successful evolutionary proposals have been developed to tackle them [23]–[28].

Coevolutionary algorithms (CAs) [29] are also able to tackle these problems. They are EAs composed of two or more populations which evolve simultaneously, allowing interactions between their individuals. In a CA, it is possible to assign different objectives or search methods to each population, trying to obtain a global solution improved by the simultaneous application of several techniques.

In recent years, coevolution has allowed many successful techniques to develop in a large number of fields. Several proposals applying CAs in classification [30], clustering [31],

Manuscript received May 10, 2011; revised November 18, 2011; accepted March 19, 2012. Date of publication April 16, 2012; date of current version September 12, 2012. This work was supported in part by the Spanish Ministry of Science and Technology under Project TIN2011-28488 and in part by the Excellence Regional Project TIC-6858. This paper was recommended by Associate Editor F. Hoffmann.

J. Derrac, I. Triguero, and F. Herrera are with the Department of Computer Science and Artificial Intelligence and the Research Center on Information and Communications Technology (CITIC), University of Granada (UGR), 18071 Granada, Spain (e-mail: jderrac@decsai.ugr.es; triguero@decsai.ugr.es; herrera@decsai.ugr.es).

S. García is with the Department of Computer Science, University of Jaén, 23071 Jaén, Spain (e-mail: sglopez@ujaen.es).

Digital Object Identifier 10.1109/TSMCB.2012.2191953

function optimization [32], training neural networks [33], or the design of ensembles [34] can be found in the literature.

CAs have also been applied in the development of data preprocessing methods to enhance the k-NN classifier. The most recent are [35] and [36], where two different approaches for data reduction (focused only on IS or combining it with feature selection, respectively) are presented. In both approaches, results improve upon those obtained by applying these data preprocessing techniques in isolation.

In this paper, we propose a new CA in a different line to the former proposals. FW and IW will be its main focus, aiming to obtain a suitable set of weights to tune the distance function while another population selects the best possible subset of instances of the training set. This tuning process will allow the k-NN rule to achieve a high classification performance, taking advantage of the existing synergy between the IS, IW, and FW techniques. We have named it Coevolution of Instance selection and Weighting schemes for Nearest Neighbor classifiers (CIW-NN).

To accomplish these tasks, three populations (one for each process) are defined within a cooperative framework. The first one performs an IS process (binary codification), aiming to select a suitable subset of instances to enhance the classification performance of the k-NN classifier. It also will try to reduce the size of the subset as much as possible, in order to increase the speed of the final classification process.

The second and third ones perform an FW and an IW process (real codification), respectively. Both are used to select the best possible weights to further increase the *leave-one-out* classification performance of the k-NN classifier. To do so, their search processes are guided by a steady-state genetic algorithm (GA) (SSGA) with a crossover operator with multiple descendants [37]. This operator is used to increase the convergence capabilities of the standard SSGA, which is a necessary improvement in the global behavior of the CA.

We have tested our proposal in a wide comparison considering several evolutionary and nonevolutionary techniques in a large number of classification domains (30 small data sets and 8 larger data sets). The results have been contrasted by using several nonparametric statistical tests for multiple comparisons, reinforcing the conclusions arrived at.

The approach presented in this paper, i.e., CIW-NN, can be used as a competent hybrid method for improving the k-NN classifier. This method combines the storage reduction and accuracy enhancement capabilities of the IS methods with the accurate definition of weights performed by IW and FW techniques, used to further improve the accuracy of the base classifier (to the best of our knowledge, this is the first method in the literature that is able to combine the three different techniques into a single approach).

Owing to cooperative coevolution and the new epoch scheme devised for managing the different populations of the model, CIW-NN is able to use cutting-edge EAs specifically adapted to the three problems, using binary and real codifications simultaneously. The joint use of the three techniques and all these elements allow CIW-NN to achieve a satisfactory performance, improving the results of all the state-of-the-art techniques considered in the study.

The rest of this paper is organized as follows. Section II gives an overview of coevolution and the data preparation techniques in the scope of this approach. Section III describes our proposal in depth. Section IV deals with the experimental framework defined. Section V shows the results obtained and discusses them. Finally, Section VI concludes the study.

II. Background: Coevolution and Evolutionary Proposals for IS and Weighting Schemes

This section covers the background information necessary to define and describe our proposal. Section II-A gives background information about coevolution and some related core issues. Section II-B describes IS as a tool to enhance the k-NN classifier. Section II-C shows the weighting schemes employed. Finally, Section II-D briefly describes some evolutionary proposals already developed to perform those techniques.

A. Coevolution: Main Trends and Key Issues

Coevolution is the field of evolutionary computation which deals with EAs that are able to manage two or more populations simultaneously. These populations coexist during the execution of the EA, interacting and evolving simultaneously.

The most important benefit of the use of coevolution is the possibility of defining several components to represent a problem and assigning them to several populations to handle each one separately. This allows the EA to employ a *divide-and-conquer* strategy, where each population can focus its efforts on solving a part of the problem. If the solutions obtained by each population are joined correctly, and the interaction between individuals is managed in a suitable way, the use of coevolution can lead to high-quality solutions, often improving those obtained by noncoevolutionary approaches.

The interaction between individuals of different populations is the core issue of coevolution techniques. In the literature, coevolution approaches are often divided into three classes, according to the type of interaction employed.

1) **Cooperative coevolution**: In this trend, each population evolves individuals representing a component of the final solution. Thus, a full solution is obtained by joining an individual chosen from each population. In this way, increases in a collaborative fitness value are shared between individuals of all the populations of the algorithm [29].
2) **Competitive coevolution**: In this trend, the individuals of each population compete with each other. This competition is usually represented by a decrease in the fitness value of an individual when the fitness value of its antagonist increases [38].
3) **Competitive–cooperative coevolution**: Both cooperative and competitive approaches can be merged, allowing the existence of a potential *arm race* among the species to improve their contributions in the associated subcomponents. This paradigm, which tries to achieve the advantages of cooperation and competition at different levels of the model, has been successfully employed in dynamic multiobjective optimization [39].

In this paper, we will focus our attention on cooperative coevolution. Its management will require the definition of an adequate problem decomposition, regarding its domain or the set of techniques employed. When this decomposition is fully defined, it will be possible to assign a baseline EA to each population, to evolve each component separately. Finally, the cooperation scheme has to be defined, analyzing the existing interdependences between the subcomponents of the model.

Although this is the general scheme when designing a cooperative coevolution approach, several key issues must be studied to understand how cooperative coevolution works and what features and disadvantages that it has.

1) The main problem of cooperative coevolution is *the loss of gradient problem* [40] in which one population comes to severely dominate the others, creating a situation where the populations have insufficient information from which to learn, due to the high degree of domination present.
2) Another problem that arises with the use of cooperative coevolution is the issue of variable interdependence. The decomposition of the domain of the problem into several parts and its assignation to the subpopulations of the model must be performed with care. Otherwise, existing interdependences between variables may severely degrade the performance of search methods.

   This issue has been studied in depth in the field of continuous optimization. Since the first cooperative coevolutionary approaches did not show satisfactory behavior in the presence of nonseparable problems, several proposals have been presented to tackle them. A representative example is shown in [32], where a *random grouping* strategy and a *weighting scheme* are presented. This framework, designed for differential evolution, has been extended to be employed with other techniques, such as particle swarm optimization [41].

   In other fields, this issue can be overcome by avoiding the breaking of interdependences. For example, an interesting way of decomposing the domain for IS problems is presented in [35]. In that study, several populations of *selectors* and a population of *combinators* are used to effectively split the domain of the IS problem, with the objective of improving the scalability of the model without harming its accuracy. In this work, a similar strategy will be followed, avoiding the breaking of interdependences not by decomposing the domain but by splitting the specific preprocessing technique assigned to each population.
3) An interesting question about the coevolutionary model is to define how the algorithm should manage its populations. Common answers are to manage them by using either a sequential scheme (update the status of the model each time a population completes a generation) or a parallel scheme (update the global status only when a generation is complete for every population). A comparison between both approaches can be found in [42].
4) These schemes can be further adjusted by using an epoch scheme [43]. Hence, a population may carry out more than one generation while the rest of the populations are stopped. This scheme can help the designer to give more importance to a given population (for example, the one with the most difficult task assigned to it) in order to keep the search balanced. If they are employed properly, epochs can help to ease the *loss of gradient problem*.
5) Researchers have also tackled the question of how to select the members to evaluate the fitness function. One way is to evaluate an individual against every single member of the other populations. However, this would consume a very high number of evaluations. To reduce this number, there are other options, such as the use of just a random individual or the use of the best individual from the previous generation [44].

In this paper, all these issues have been considered and tackled during the designing process of CIW-NN. More information can be found in Section III.

B. IS

IS is one of the main data reduction techniques for which many proposals have been developed (see [12] or [45] for a recent review). Its goal is to isolate the smallest set of instances which enable a data mining algorithm to predict the class of a query instance with the same or better proficiency than using the initial data set [16]. By minimizing the data set size, the space complexity and computational costs of the subsequent data mining algorithms are reduced, improving their generalization capabilities.

IS can be defined as follows: Let X be an instance where $X = (x_1, x_2, \ldots, x_M, x_c)$, with X belonging to a class c given by X_c, and an M-dimensional space in which x_i is the value of the ith feature of the sample X. Then, let us assume that there is a training set TR which consists of N instances, and a test set TS composed of T instances. Let $RS \subseteq TR$ be the subset of selected samples that result from the execution of an IS algorithm; then, we classify a new pattern T from TS from a data mining algorithm acting over the instances of RS.

We focus our efforts on enhancing the 1-NN rule. The reason for not employing a value $k > 1$ for the k-NN is to give the classifier the greatest possible sensitivity to noise during the reduction process. In this manner, an evolutionary IS algorithm can better detect the noisy instances and the redundant ones presented in the training set.

C. Weighting Schemes

In this section two different weighting schemes, i.e., FW and IW, will be reviewed.

a) FW: FW is a successful approach used to improve the k-NN classifier [11]. The FW methods' main objective is to reduce the sensitivity of redundant or noisy features in the k-NN rule by modifying its distance function.

The most well-known dissimilarity measure for the k-NN rule is the Euclidean distance [(1), where X and Y are two instances and M is their number of features]. It has been widely used in the instance-based learning field [9].

$$EuclideanDistance(X, Y) = \sum_{i=0}^{M} \sqrt{(x_i - y_i)^2}. \quad (1)$$

FW methods often extend this equation to apply different weights to each feature (W_i) which modify the way in which the distance measure is computed

$$FWDist(X,Y) = \sum_{i=0}^{M} W_i \cdot \sqrt{(x_i - y_i)^2}. \quad (2)$$

This technique has been widely used in the literature. To the best of our knowledge, the most complete study performed can be found in [11], where a review of several FW methods for lazy learning algorithms is presented (with most of them applied to improve the performance of the 1-NN rule).

A large number of FW techniques for improving the k-NN rule have been proposed [11], [46]. The most well known form the family of *Relief-based* algorithms. The Relief algorithm has been widely studied and modified, producing some interesting versions such as that in [47].

b) IW: The second interesting weighting scheme is IW. This scheme consists of applying weights to the instances of the training set, modifying the distance measure between them and any other instance (the following equation, where $IW(X)$ is the weight assigned to the training instance X):

$$IWDist(X,Y) = IW(X) \sum_{i=0}^{M} \sqrt{(x_i - y_i)^2}. \quad (3)$$

The concrete definition of the weights differs in each approach, but most of the existing ones are focused on modifying the way in which distances are measured, depending on the positions of the instances in the training set (a representative example that appeared recently is [17]).

Other interesting approaches apply the weights in a lazy way: Weights are assigned to instances only when the query instance (i.e., a test instance) has been presented to the classifier. In this way, only instances which are in the immediate environment of the query instance are weighted, thus performing an *ad hoc* local modification directly focused on the concrete query instance presented [10].

Although the number of existing proposals for IW is less than that for FW, several interesting approaches have appeared in recent years, mostly focused on the application of weights to find a suitable local metric to improve the generalization accuracy of the basic 1-NN [48].

D. Existing Evolutionary Approaches Used to Improve the k-NN Rule

In recent years, EAs have been widely employed to carry out data preparation tasks, improving the behavior of the k-NN. This section will review some interesting examples in the scope of this study, most of them applied to IS tasks.

The first use of EAs in IS can be found in [24]. Kuncheva applied a GA to select a reference set for the k-NN rule. Her GA maps the training set onto a chromosome structure, using a binary representation and computing as the fitness function the error rate of the k-NN rule.

In [23], a complete study of the use of EAs in IS is made, highlighting four EAs to complete this task: Cross-generational elitist selection, Heterogeneous recombination, and Cataclysmic mutation (CHC) adaptive search algorithm [49], SSGA, generational GA, and population-based incremental learning. They conclude that EAs outperform classical algorithms in both reduction rates and classification accuracy, highlighting CHC as an outstanding method for this task. Following this line, other successful proposals have appeared recently [26].

In [50], a GA is used to learn continuous feature weights for the k-NN classifier, by defining five genetic operators and a fitness function based on the number of misclassified training instances and their relevance. Furthermore, it is possible to find other approaches that combine several techniques in the same method. For example, [25] is a representative proposal combining IS and feature selection to improve k-NN classifiers. FW and IS are also tackled simultaneously in [51].

Cooperative coevolution has also been used to improve the k-NN rule in [35], integrating several populations of selectors and a population of combinators to effectively split the domain of the IS problem, and [36], in which cooperative coevolution is adopted as a tool to integrate several binary preprocessing techniques in the coevolutionary model (by defining a multi-classifier composed of three 1-NN classifiers which are tuned by each population), which are representative examples. However, these two approaches neither define specific mechanisms to adjust the search performed in each population (different basic EAs and codification, fine-tuned operators, and so on) nor consider the definition of weighting schemes to further improve the classification accuracy.

III. CIW-NN

In this section, we describe CIW-NN in depth. We show the details of the architecture of the coevolutionary model and its basic components, justifying the decisions taken during its development. Section III-A shows the scheme of populations of CIW-NN. Section III-B gives an overview of the full coevolutionary model and describes the basic techniques used to conduct the search. Section III-C describes how the cooperation between individuals belonging to different populations is achieved, through the fitness function. Finally, Section III-D states how the fitness value is assigned to each chromosome.

A. Population Scheme

CIW-NN is composed of three populations, which coexist and evolve simultaneously. We denote each one by the name of its assigned task. Therefore, our model is composed of an IS population, an IW population, and an FW population, which will follow a sequential scheme of cooperation.

In order to characterize and describe them, several aspects of their structure and behavior must be discussed.

1) **Scope**: Each population is focused on optimizing either instances or features.
2) **Codification**: Depending on the concrete assessing task performed, the individuals of each population will employ binary (0, 1) or real ([0, 1]) codification. This feature will define the kind of basic search method which the

TABLE I
CIW-NN POPULATION'S CHARACTERISTICS

Topic	IS population	IW population	FW population
Scope	Instances	Instances	Features
Codification	Binary	Real	Real
Granularity	Individual	Class	Individual
Epoch length	Simple	Multiple	Multiple
Objective	Acc./Red.	Accuracy	Accuracy

population will carry out and also has a strong effect on the difficulty of the search task itself, due to real coded search spaces usually being wider and harder to explore.

3) **Granularity**: CIW-NN uses two schemes of assignation of weights. Individual weights (one for each instance/feature) are assigned to IS and FW chromosomes, whereas class weights, shared by instances of the same class, are assigned to IW chromosomes.
4) **Epoch length**: CIW-NN defines how the evolution process of its populations will be scheduled, by assigning epochs of different lengths: Simple, i.e., one generation per cycle of the global model, or Multiple, considering more than one generation. In this way, CIW-NN equalizes the number of evaluations spent by each population.
5) **Objective**: This refers to the objective that each population pursues. A population can cope with maximizing the accuracy obtained by the classifier, or to simultaneously maximize this accuracy and the reduction rate, i.e., the ratio between the number of instances discarded and the ones that composed the original training set.

Table I summarizes the setup of each population. As can be seen in this table, the objective of obtaining a reduced subset is assigned to the IS population, while the rest tune the way in which distances to the instances are computed, accomplishing the objective of increasing the accuracy of the classifier.

Owing to this structure, our model is expected to achieve reduction rates close to those obtained by the most successful evolutionary IS techniques and simultaneously obtain better results in accuracy due to the double tuning process performed by IW and FW populations. Furthermore, the tuning process performed by the weighting populations can positively influence the behavior of IS, for example, by selecting weights to make up instances that, without this weighting process, could be marked as irrelevant—or even noisy—by the selection process. Fig. 1 symbolizes the feedback between populations existing in CIW-NN.

It is important to note that we have not considered the use of feature selection in CIW-NN (for example, as a new population of the model). The reason for this is that the weights of the FW population can simulate this behavior just by using weights very near to 0.0 or 1.0. The rejection of the use of feature selection prevents CIW-NN from achieving even greater reduction rates than those it achieves as it is currently defined. However, this is compensated for by the increase of the search space of the features, which should lead CIW-NN to tune the distance function better. This capability should help in increasing the accuracy of the model.

On the other hand, one should not expect similar behavior if weights were applied to every instance to simulate IS. This is because, in most of the standard classification problems, the number of instances is far higher than the number of features. Therefore, a reasonable way to work with instances would be to apply a procedure to quickly select them, and another one is to tune them in a fast and effective way (for example, grouping them by their class). This is the reason behind the decision to use class weights in the IW population.

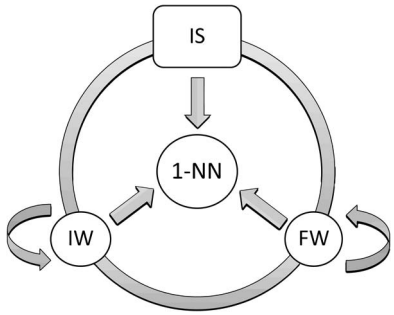


Fig. 1. Evolutionary cycle of CIW-NN. IS, FW, and IW populations evolve simultaneously, spending an epoch in turns. IS epochs only spend one generation, while FW and IW epochs spend several. Individuals of every population cooperate through the fitness function (a 1-NN classifier).

```
1: Generate ISPopulation,FWPopulation and IWPopulation Randomly
2: Select initial best performing individuals (ISBest, IWBest, FWBest)
3: while evaluations < max_evaluations do
4:    epoch(ISPopulation)
5:    epoch(IWPopulation)
6:    epoch(FWPopulation)
7:    ISBest = Best(ISpopulation)
8:    IWBest = Best(IWpopulation)
9:    FWBest = Best(FWpopulation)
10: end whileISBest, IWBest, FWBest
```

Fig. 2. Coevolutionary model.

B. Coevolutionary Model

The coevolutionary model consists of the three populations carrying out their respective search processes at the same time: In each cycle, each population performs a fixed number of generations, depending on their concrete setup, but their state is not updated until the generations of the rest of the populations have finished. Populations with Simple epoch length (IS population) will perform only a single generation, while populations with Multiple epoch length (IW and FW populations) will perform several generations. When the fixed number of evaluations runs out, the best individuals found are taken as the output of the method. Then, they are used to build a final preprocessed training set, which will be ready to be used by a 1-NN classifier to classify the test set or any new example. Fig. 2 shows a pseudocode of the full model.

CIW-NN only selects one individual per population (the best) as a collaborator. Although other schemes, such as selecting a set of collaborators per population, could be defined, the employment of just one collaborator has a unique advantage: The evaluation of any new individual only consumes one evaluation of the fitness function. By contrast, employing any other scheme would lead to a quadratic increase in the number of evaluations needed to characterize a new individual

(if Z collaborators per each of the three populations are selected, a new individual will require Z^2 evaluations to consider all the possible combinations). Given the nature of the problems tackled by CIW-NN, which are characterized by a costly fitness function, computationally speaking (more costly, by far, than, for example, the ones usually considered in function optimization problems; see Section III-C), this decrease in the evaluation requirements is indispensable.

The use of a parallel scheme of coevolution, where the global status of the model is updated only when a generation is complete for every population, and Simple and Multiple epoch schemes allows CIW-NN to effectively combine different kinds of EAs. In this way, different basic evolutionary techniques can be assigned to each population, selecting for each task and codification the most suitable technique.

Concretely, CIW-NN uses an adapted version of the CHC algorithm [49] in the IS population, whose reliability in its application to IS problems has already been studied in [23]. In that study, the authors concluded that the CHC algorithm is a very suitable evolutionary approach to perform IS processes in order to enhance the performance of the 1-NN classifier.

In FW and IW populations, CIW-NN uses an SSGA with multiple descendants [37]. We have selected it since it has shown a good performance when applied to continuous optimization problems with a high number of variables. Furthermore, the use of multiple descendants gives a strong convergence capability to the SSGA, which is the most desired quality for the search process of the FW and IW populations.

Both basic methods evolve each population within the evolutionary cycle. Since the number of evaluations spent in one generation of the IS population can be much greater than that in one generation of the FW and IW populations (CHC is a generational GA which can spend as many evaluations as the population size to perform a generation, whereas the SSGA only spends a much smaller amount per generation), CIW-NN introduces the use of a Multiple epoch scheme. In this way, $NGens$ generations are carried out for FW and IW populations in each epoch, whereas only one is carried out for the IS population. This will help to equalize the number of evaluations spent, regardless of the search method used.

In order to adjust the behavior of both search methods, several modifications to the original algorithms have been considered.[1] The main drawback of the application of CHC in the IS problem is that the efficiency of its fitness function depends on the phenotype of the chromosome. If there are many 1's in the binary chromosome, many instances will be selected, increasing the cost of the 1-NN classification performed to compute its fitness value (Section III-C).

Therefore, we have applied two modifications to the original algorithm to increase the speed of the IS population.

1) We have modified the definition of the half uniform crossover (HUX) operator. When a gene representing an instance is going to be set from 0 to 1 by the crossing procedure, it is only set to 1 with a defined probability ($prob0to1$ parameter). No modifications are applied to changes from 1 to 0. For example, if one chromosome, 1100000000, and another chromosome, 1111111111, are crossed by the HUX operator, the offsprings may be 1111110000 and 1100001111. In the same scenario, a run of our HUX modified operator, with a probability of change $prob0to1 = 0.5$, would give the offsprings 1100110000 and 1100001111 as the output. Fig. 3 shows its application.
2) The initialization of the individuals is made randomly, but only a small fixed number of genes are set to 1. Therefore, in the initialization of a chromosome, each gene has a probability *prob1* to be set to 1.

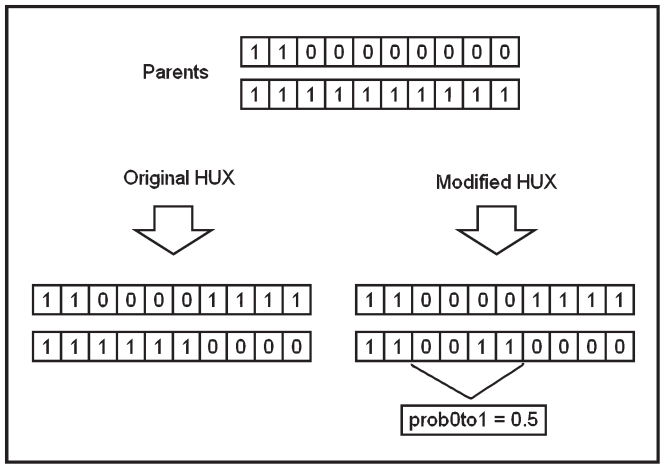


Fig. 3. HUX crossover operator exchanges exactly half of the nonmatching alleles, selected randomly. In our modified version of HUX, an allele valued with 1 has a probability $prob0to1$ of maintaining its value when it is selected to be exchanged.

The $prob0to1$ parameter does not have a great impact on the results if it is kept in the interval (0.2–0.5). A value lower than 0.2 may bias the search, making it very difficult for CHC to preserve the quantity of 1's in the chromosomes. This may force the algorithm to produce solutions with high reduction rates but very low performance in accuracy due to the impossibility of selecting enough instances to represent the initial training set properly. On the other hand, a value higher than 0.5 will diminish the effect of the operator, producing solutions with lower reduction rates. Consequently, we have defined $prob0to1 = 0.25$ as an optimal setup. The *prob1* also does not have a great impact on the results, as long as it is set to a low value. Defining a value of 0.5 will be the same as defining just a random initialization. Thus, this value has to be lower. In our experiments, we have found that $prob1 = 0.25$ is an optimal setup for this value, which helps CHC to quickly obtain reduced solutions without biasing the search process.

The SSGA has the following features.

1) Initialization of individuals is made randomly, assigning to each gene weights valued in the interval [0, 1].
2) Binary tournament is used to select the parents (two individuals are randomly taken from the population. Then, the one with the best fitness value is selected. This procedure is carried out twice to obtain the two parents required).
3) The offspring is obtained using a crossover operator with multiple descendants. It consists of repeatedly

[1] A wide description of them can be found at http://sci2s.ugr.es/ciw-nn.

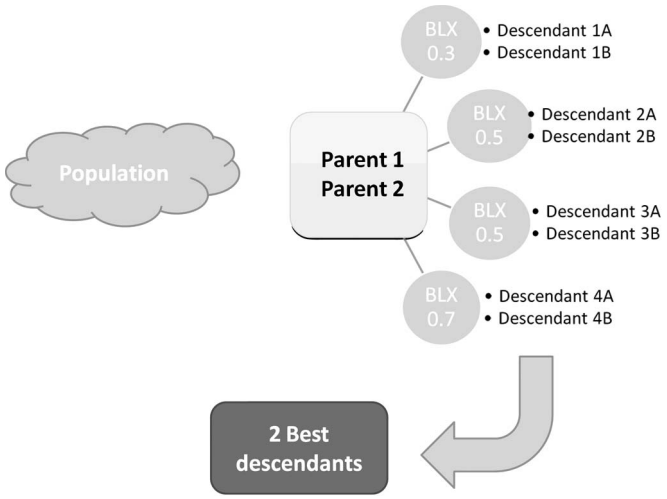


Fig. 4. Scheme of application of the 2BLX0.3–4BLX0.5–2BLX0.7 multiple-descendant crossover operator.

applying one or more standard crossover operators to obtain several new individuals (six or eight are common numbers). Then, the best two are selected as the new offsprings. From all the options suggested in [37], we have selected the blend crossover operator (BLX-α), applying it four times with values 0.3, 0.5, 0.5, and 0.7 for α (2BLX0.3–4BLX0.5–2BLX0.7). Fig. 4 shows its application.

4) A mutation operator is applied to every descendant obtained during the multiple-descendant crossing process. Following [37], we have used the nonuniform mutation operator [52]. Mutation probability is set to a low value, i.e., 0.05 per chromosome, to avoid harming the convergence capabilities of the crossover operator.
5) The two individuals of the current population with the worst fitness value are replaced by the new offsprings.

C. Cooperation in CIW-NN: The Fitness Function

In CIW-NN, the cooperation between individuals is achieved by merging three chromosomes (one from each population: IS, IW, and FW). Basically, they are used to generate a preprocessed version of the training set (i.e., a reduced version of the training set by the application of the IS process, and an assignation of weights to instances and features by the application of the FW and IW processes) whose quality will be evaluated by a 1-NN classifier.

Therefore, we define the fitness function of CIW-NN [the following equation, where J, K, and L are the three chromosomes selected] as the accuracy rate estimated when classifying the original training set, using the preprocessed one as a reference set and using leave-one-out as a validation scheme

$$Fitness(J,K,L) = AccuracyRate(J,K,L). \quad (4)$$

When a new chromosome is evaluated, two collaborators, from the other populations, are merged with it to create a full solution. The process performed to obtain the preprocessed training set from the original one is the following.

1) Instances marked as "0" by the IS chromosome are removed. Thus, only instances marked as "1" will remain in the preprocessed set.
2) Weights described by the FW chromosome are assigned.
3) Weights described by the IW chromosome are assigned to the remaining instances, depending on their class.

As a result of these operations, the computation of the distance measure in the 1-NN classifier is performed as is described in (5), where X is an instance of the preprocessed set, Y is a new instance to classify (from the original training set or from the test set), and $IW_{c(X)}$ is the weight assigned by the IW chromosome

$$Distance(X,Y) = \gamma \cdot \left(1.0 - IW_{c(X)}\right) \cdot FWDist(X,Y) + (1.0-\gamma) \cdot FWDist(X,Y) \quad (5)$$

where $\gamma \in [0,1]$ is a weighting value for controlling the impact of the IW weights in the Euclidean weighted distance (2). Consequently, we have the following.

1) The distances computed from instances belonging to classes marked with maximum weights ($IW_{c(X)} = 1.0$) will be very small (a $(1.0-\gamma)$ factor of their former value).
2) The distances computed from instances marked with minimum weights ($IW_{c(X)} = 0.0$) will not change.
3) The remaining possible values will keep the distance computed within this range.

Following this scheme, we allow the IW population to set weights which highlight the appearance of certain classes in the domain, diminishing the importance of the rest. The distances computed from instances belonging to the former classes will be very low, thus increasing the chances of selecting them as the nearest neighbors of a new test instance. Furthermore, the inclusion of the γ weight allows the appearance of very low final weights (very near to 0.0) to be avoided, which may nullify the distance measure, degrading the behavior of the classifier.

With the simultaneous application of IS and both weighting schemes, the preprocessed subset can be tuned to obtain the most accurate possible reference set. IS chromosomes will select only those instances which are truly relevant in the training set, whereas FW weights will emphasize those features which better discriminate the examples with respect to their class. Finally, IW weights will increase or decrease the magnitude of the distance from every instance, depending on its class, modifying its relevance inside the domain, further improving the global accuracy of the classifier.

A final consideration about the fitness function of CIW-NN must be noted: The computation of the $AccuracyRate$ involves the classification of the entire training set by the 1-NN classifier. This is a costly operation ($O^{(N*S)}$, where N is the size of the training set and S is the number of instances selected by the IS chromosome) and computationally heavier than the fitness function usually employed as a benchmark in existing approaches for function optimization. However, this cost will be alleviated as the search processes progress, as long as the IS chromosomes reduce the number of instances selected.

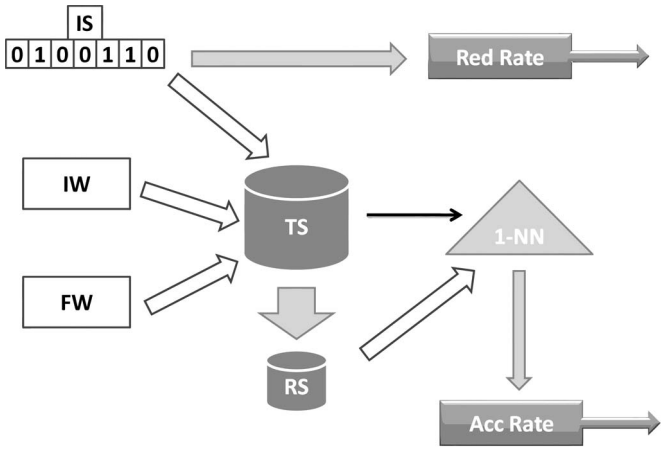


Fig. 5. Example fitness assignment for an IS individual. $RedRate$ is computed directly. To compute $AccRate$, IS, FW, and IW chromosomes are applied to the training set TS, obtaining the reference set RS, which is employed by the 1-NN classifier to estimate accuracy.

D. Fitness Assignment

As we mentioned before, the populations of CIW-NN have different objectives, depending on their task. Therefore, two different methods for fitness assignment need to be defined.

The fitness value for IS individuals must pursue both reduction and accuracy objectives. To do so, we follow the proposal given in [23]. Cano *et al.* defined $AccRate$ as the accuracy achieved by the 1-NN rule over the training set, using the currently selected subset as a reference and leave-one-out as a validation scheme. They also defined $RedRate$ as the reduction rate achieved over the currently selected instances, and a weighting factor α, to adjust the strength of each term in the resulting fitness value. The following equation defines it, where J is an IS chromosome to be evaluated:

$$Fitness(J) = \alpha \cdot AccRate(J) + (1-\alpha) \cdot RedRate(J). \quad (6)$$

Following the recommendations given in [23], CIW-NN employs a value $\alpha = 0.5$, which should offer an adequate tradeoff between accuracy and reduction.

Obtaining a fitness value for the IW and FW is straightforward, since their objective is to maximize only the accuracy of the classifier. The following equation, where K is a chromosome belonging to FW or IW populations, can be defined by considering only the $AccRate$ term of the last equation, keeping their former meaning:

$$Fitness(K) = AccRate(K). \quad (7)$$

These equations are used to assign a fitness value to any chromosome of CIW-NN. When the fitness function is computed by using a chromosome in combination with its two collaborators, the fitness value obtained is assigned as its $AccRate$. On the other hand, if the chromosome belongs to the IS population, its $RedRate$ can be computed directly from the chromosome itself. Fig. 5 shows an example of a fitness assignment for an IS individual.

TABLE II
SUMMARY DESCRIPTION OF SMALL DATA SETS

Data set	#Ex.	#At.	#Cl.	Data set	#Ex.	#At.	#Cl.
Australian	690	14	2	Monk-2	432	6	2
Balance	625	4	3	Movement	360	90	15
Bands	539	19	2	New Thyroid	215	5	3
Breast	286	9	2	Pima	768	8	2
Bupa	345	6	2	Saheart	462	9	2
Car	1728	6	4	Sonar	208	60	2
Cleveland	303	13	5	Spectfheart	267	44	2
Contraceptive	1473	9	3	Tae	151	5	3
Dermatology	366	34	6	Tic-tac-toe	958	9	2
German	1000	20	2	Vehicle	846	18	4
Glass	214	9	7	Vowel	990	13	11
Hayes-roth	160	4	3	Wine	178	13	3
Housevotes	435	16	2	Wisconsin	699	9	2
Iris	150	4	3	Yeast	1484	8	10
Lymphography	148	18	4	Zoo	101	16	7

TABLE III
SUMMARY DESCRIPTION OF LARGE DATA SETS

Data set	#Ex.	#At.	#Cl.	Data set	#Ex.	#At.	#Cl.
Abalone	4174	8	28	Page-blocks	5473	10	5
Banana	5300	2	2	Phoneme	5404	5	2
Chess	3196	36	2	Segment	2310	19	7
Marketing	8993	13	9	Splice	3190	60	3

IV. EXPERIMENTAL FRAMEWORK

This section describes the experimental framework designed to test CIW-NN.[2] Section IV-A presents the classification data sets used. Section IV-B summarizes the algorithms selected for the comparison and their relevant parameters. Section IV-C describes the performance measures employed to evaluate CIW-NN. Finally, Section IV-D discusses the tests applied in the statistical comparisons performed.

A. Classification Problems

To check the performance of CIW, we have selected a set of 38 classification data sets. These are well-known problems in the area, taken from the KEEL-data-set repository[3] [53]. Tables II and III summarize their main characteristics. For each data set, we provide its number of examples (#Ex.), attributes (#At.), and classes (#Cl.).

The data sets considered are partitioned by using the ten-fold cross-validation (10-fcv) procedure, and their values are normalized in the interval [0, 1] to equalize the influence of attributes with different range domains. In addition, instances with missing values have been discarded before the execution of the methods over the data sets.

B. Comparison Methods

Several classification methods, evolutionary and nonevolutionary, have been selected to perform an exhaustive study of the capabilities of CIW-NN.

1) **1-NN**: The 1-NN rule is used as a baseline limit of performance which most of the methods should supersede.

[2]The Java code of CIW-NN is available at http://sci2s.ugr.es/ciw-nn.
[3]http://www.keel.es/datasets.php.

2) **IS-CHC, FW-SSGA, and IW-SSGA**: These methods follow exactly the same setup as the populations of CIW-NN, except that the $AccRate$ of their fitness function is computed separately. Consequently, only the IS-CHC performs a reduction process. Moreover, comparison with IS-CHC is particularly interesting due to it being recommended as the best performing IS method in [23].
3) **Steady-state memetic algorithm (SSMA)**: An SSMA specifically designed for prototype selection. This evolutionary IS includes a meme optimization mechanism (a local search procedure) that is able to improve the accuracy achieved by the SSGA and to avoid premature convergence. Moreover, it offers a high reduction capability and good behavior when tackling large problems [26].
4) **Prototype weighting (PW), class weighting (CW), and class and prototype weighting (CPW)**: Three gradient-descent-based algorithms developed with the aim of minimizing a performance index that is an approximation of the *leave-one-out* error over the training set. Weights may be specified for each instance (PW) and for each combination of feature and class (CW) or both (CPW) [48].
5) **Weighted distance nearest neighbor (WDNN)**: A novel IW method which searches iteratively, in each training instance, for the best weight to minimize the *leave-one-out* error over the training set. A weight of 0.0 can be assigned to any instance, which means that it is discarded. Thus, this method can be regarded as a simultaneous IS and IW method. Consequently, reduction rates can be computed for it [17].
6) **Tabu search for KNN (TS/KNN)**: A tabu-search-based method for simultaneous feature selection and FW, whose solutions encode the current set of features selected, the current set of weights assigned to features, and the best value of k found for the k-NN classifier [54]. Although this method reduces the size of the training set by selecting features, this reduction is too small for it to be considered in the comparison with the rest of the methods.
7) **ReliefF**: The first *Relief-based* method adapted to perform the FW process [47]. Weights computed in Relief are not binarized to 0, 1. Instead, they are used as final weights for the k-NN classifier. This method was noted as the best *performance-based* FW method in [11].
8) **Mutual information (MI)**: MI between features can be used successfully as a weighting factor for k-NN-based algorithms. This method was marked as the best *preset* FW method in [11].
9) **Global optimization of feature weighting and instance selection using GA for case-based reasoning (GOCBR)**: A GA for simultaneous IS and FW. Weights are represented by binary chains, using binary codification [51]. This method was not designed with the aim of obtaining the most reduced subset possible; thus, its reduction power is not competitive. Therefore, their reduction rates will not be considered.

Many different configurations have been established for each method. In our experimental study, we have used the parameters defined in the reference, where they were originally described. Table IV presents them.

TABLE IV
PARAMETER SPECIFICATION FOR THE METHODS OF THE STUDY

Algorithm	Ref.	Parameters
CIW-NN	-	Evaluations: 10000, Populations size (IS, FW, IW): 50, Crossover operator: 2BLX0.3-4BLX0.5-2BLX0.7,γ :0.8 Mutation prob.: 0.05 per chromosome, $prob1$: 0.25, $prob0to1$: 0.25, Epoch length: 40 evals., α: 0.5
IS-CHC	-	Evaluations: 10000, Population size: 50, α: 0.5, $prob0to1$: 0.25, $prob1$: 0.25
FW-SSGA	-	Evaluations: 10000, Population size: 50, Crossover operator: 2BLX0.3-4BLX0.5-2BLX0.7, Mutation prob.: 0.05 per chromosome
IW-SSGA	-	Evaluations: 10000, Population size: 50, Crossover operator: 2BLX0.3-4BLX0.5-2BLX0.7, Mutation prob.: 0.05 per chromosome
SSMA	[26]	Evaluations: 10000, Cross prob.: 0.5 per bit Population size: 30, Mutation prob.: 0.001 per bit
PW	[48]	β: Best in $[0.125, 128]$, ρ: Best in $[0.1, 0.001]$, ϵ: 0.001, Iterations: 1000
CW	[48]	β: Best in $[0.125, 128]$, μ: Best in $[0.1, 0.001]$, ϵ: 0.001, Iterations: 1000
CPW	[48]	β: Best in $[0.125, 128]$, μ: Best in $[0.1, 0.001]$, ρ: Best in $[0.1, 0.001]$, ϵ: 0.001, Iterations: 1000
WDNN	[17]	Iterations: 10
TS/K-NN	[54]	Evals.: 10000, M: 10, N: 2, P: ceil($\sqrt{\#Features}$)
ReliefF	[47]	K value: Best in $[1, 20]$
MI	[11]	It has no parameters to be fixed
GOCBR	[51]	Evaluations: 10000, Population size: 100, Crossover prob.: 0.7, Mutation prob.: 0.1

C. Performance Measures

We have selected the following performance measures.

1) **Accuracy**: It 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 [1], [4].
2) **Kappa**: It is an alternative to the accuracy rate, a method, known for decades, that compensates for random hits [55] in the same way as the Area Under the ROC curve measure. Cohen's kappa measure can be obtained using the expression

$$kappa = \frac{N \sum_{I=1}^{c} x_{ii} - \sum_{i=1}^{c} x_{i.} x_{.i}}{N^2 - \sum_{i=1}^{c} x_{i.} x_{.i}} \tag{8}$$

where x_{ii} is the cell count in the main diagonal, N is the number of examples, c is the number of class values, and $x_{.i}$ and $x_{i.}$ are the column and row total counts, respectively. Kappa ranges from -1 (total disagreement) through 0 (random classification) to 1 (perfect agreement). For multiclass problems, it is a very useful, yet simple, metric for measuring the accuracy of the classifier while compensating for random successes.
3) **Reduction**: The reduction rate is defined as the ratio of data selected by the algorithm. It has a strong influence on the efficiency of the solutions obtained, due to the cost of the final classification process performed by the 1-NN classifier $(O(N^2 \cdot M))$.
4) **Time**: The simplest way to measure the practical efficiency of a method. We will analyze the average time elapsed (in seconds) by every method, considering both training and classification phases.

TABLE V
AVERAGE RESULTS OBTAINED IN THE COMPARISON BETWEEN CIW-NN AND EVOLUTIONARY PROPOSALS FOR k-NN-BASED CLASSIFICATION

	Accuracy		Kappa		Reduction	Time	Ranks (Acc.)		Ranks (Kap.)	
Method	Training	Test	Training	Test	Red.	Time (s.)	Fried.	F. Alig.	Fried.	F. Alig.
CIW-NN	78.04±1.64	80.09±4.80	0.6018±0.0242	0.6192±0.0974	0.9189	96.74	2.28	56.18	2.59	63.08
IS-CHC	79.55±1.30	78.00±5.64	0.6169±0.0242	0.5810±0.1112	0.9547	86.50	3.81	94.31	3.46	92.50
FW-SSGA	83.18±0.97	78.83±5.08	0.6931±0.0190	0.6111±0.0975	-	243.71	3.50	82.90	3.08	73.14
IW-SSGA	79.25±0.94	77.56±5.11	0.5780±0.0195	0.5474±0.1001	-	243.91	3.46	91.83	4.11	116.46
SSMA	80.87±1.14	77.44±5.92	0.7016±0.0239	0.5768±0.1067	0.9580	90.94	3.38	91.75	3.41	93.80
1-NN	74.61±1.03	74.69±5.36	0.5294±0.0206	0.5297±0.1057	-	-	4.55	126.01	4.32	122.00

D. Statistical Tools for Analysis

In our experimental study, we use hypothesis testing techniques to provide statistical support for the analysis of the results. Concretely, we use nonparametric tests, due to the fact that the initial conditions that guarantee the reliability of the parametric tests may not be satisfied, causing the statistical analysis to lose credibility [56].

Throughout the study, we will use the Friedman and Friedman aligned-ranks tests to detect statistical differences among the methods. Holm, Hochberg, and Finner *post hoc* procedures will be used to find out which methods are distinctive among the $1 * n$ comparisons performed [56]. Moreover, the ranks obtained will be analyzed graphically, depicting the best performing algorithms as those with lower ranks.

More information about those statistical procedures specifically designed for use in the field of machine learning can be found at the SCI2S thematic public Web site on *Statistical Inference in Computational Intelligence and Data Mining*.[4]

V. RESULTS AND ANALYSIS

In this section, we detail the different experimental studies carried out with CIW-NN. In particular, our aims are as follows:

1) to analyze the benefits of coevolution when integrating several techniques, comparing CIW-NN with other evolutionary methods in isolation (Section V-A);
2) to compare CIW-NN with classical and recent weighting methods for k-NN-based classification (Section V-B);
3) to test the performance of CIW-NN when the size of the problem increases (Section V-C);
4) to show the convergence process of CIW-NN and the cooperation process among populations (Section V-D).

For the sake of simplicity, we only include average results, whereas the complete results can be found elsewhere.[5] These average results are computed through a 5 × 10-fold cross-validation procedure, which means that every algorithm has been run ten times per data set (one for each partition), and this process has been repeated five times, averaging the results obtained. This will allow us to draw strong conclusions, reducing the danger of being mislead by outliers or random successes in the results of the classifiers.

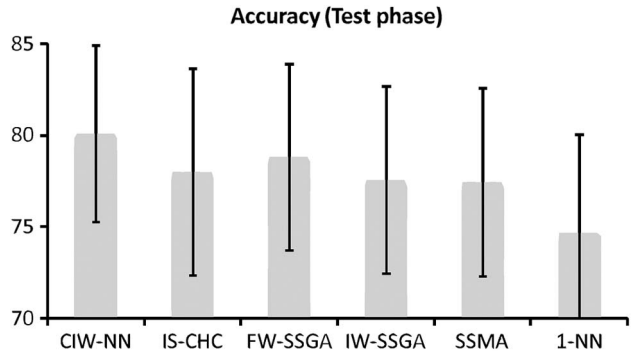


Fig. 6. Graphical comparison of accuracy in the test phase between CIW-NN and evolutionary proposals for k-NN-based classification.

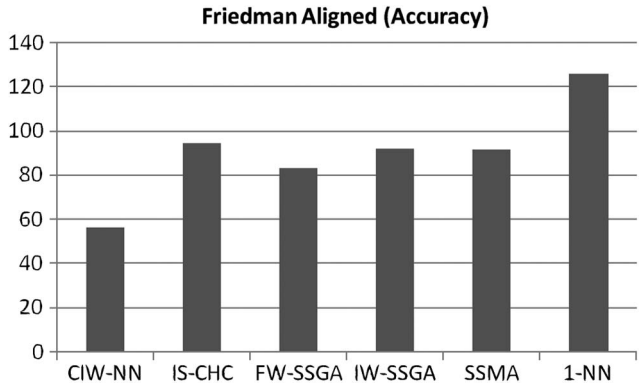


Fig. 7. Rankings computed by Friedman aligned-ranks procedures by using the accuracy measure.

A. Comparison Between CIW-NN and Evolutionary Proposals for k-NN-Based Classification

The coevolution abilities of CIW-NN can be stressed when it is compared with its basic components. In this paper, the three evolutionary techniques employed in CIW-NN (IS-CHC, FW-SSGA, and IW-SSGA) will be applied in isolation. Furthermore, we consider the 1-NN classifier as a basic reference, and SSMA, an advanced method for IS which incorporates a competent local optimizer to improve the search process.

Table V shows the results obtained in the 30 small data sets. Fig. 6 emphasizes the accuracy results of the test phase graphically, showing mean accuracy and standard deviations.

[4]http://sci2s.ugr.es/sicidm/.

[5]http://sci2s.ugr.es/ciw-nn. On this Internet site, it is possible to find results detailed for each data set and performance measure, including several graphical comparisons summarizing the results achieved in the experiments and depicting the rankings obtained by the algorithms in each application of Friedman and Friedman aligned-ranks procedures. It also contains the results of the application of Holm, Hochberg, and Finner *post hoc* procedures. Furthermore, several related studies about the behavior of our method (a sensitivity analysis of parameters, the selection of suitable crossover operators with multiple parents, advanced schemes of combination of the different preprocessing techniques, selection of an optimal value for the k parameter of k-NN, and so on) can also be found there.

TABLE VI
AVERAGE RESULTS OBTAINED IN THE COMPARISON BETWEEN CIW-NN AND WEIGHTING METHODS FOR k-NN

	Accuracy		Kappa		Reduction	Time	Ranks (Acc.)		Ranks (Kap.)	
Method	Training	Test	Training	Test	Red.	Time (s.)	Fried.	F. Alig.	Fried.	F. Alig.
CIW-NN	78.04±1.64	80.09±4.80	0.6018±0.0242	0.6192±0.0974	0.9189	96.74	3.16	79.73	3.38	88.75
TS/KNN	79.61±1.34	75.76±5.61	0.6205±0.0238	0.5393±0.0953	-	503.07	4.76	136.70	5.10	147.86
PW	83.59±1.05	78.34±5.39	0.6651±0.0222	0.5842±0.1083	-	1.11	4.63	125.05	4.96	128.03
CW	79.42±1.28	77.56±5.44	0.6230±0.0218	0.5797±0.1070	-	0.57	5.93	147.33	5.48	133.41
CPW	82.28±1.49	78.42±5.31	0.6289±0.0223	0.5851±0.1085	-	0.89	4.33	120.15	5.08	128.25
ReliefF	75.75±2.32	75.78±5.51	0.5484±0.0240	0.5445±0.1079	-	6.07	5.11	140.35	5.10	143.93
MI	74.22±2.15	72.97±5.43	0.5428±0.0260	0.5189±0.1115	-	0.16	6.78	190.48	5.93	165.30
GOCBR	85.89±0.90	77.19±5.84	0.7375±0.0246	0.5665±0.1138	-	241.87	5.50	144.56	5.23	141.46
WDNN	85.42±1.61	77.52±5.71	0.7256±0.0246	0.5583±0.1121	0.896	31.82	4.76	135.13	4.71	142.48

The results obtained show that CIW-NN is the best performing algorithm in the test phase. Moreover, all the techniques selected are able to improve the baseline performance of the 1-NN classifier. Fig. 7 plots the rankings obtained by the Friedman aligned-ranks test with the accuracy measure, showing the differences found among the different methods graphically.

After computing the ranks, the Friedman and Friedman aligned-ranks procedures obtained p-values of 0.00032 and 0.000097 for accuracy and p-values of 0.00274 and 0.000064 for kappa, respectively. Thus, the two tests detected significant differences (final p-values computed by the *post hoc* methods are reported on the associated Internet site).

Using these results, we can conclude the following.

1) CIW-NN offers the best results in accuracy and kappa measures (in the test phase). Only FW-SSGA is able to obtain close results (and only by the kappa measure).
2) The reduction rates achieved by CIW-NN are close to those of IS-CHC and SSMA. This is a good result if we recall that the IS population of CIW-NN (the only one which aims to reduce the data set) only has a third of the total evaluations spent by the coevolutionary model.
3) CIW-NN achieves the best results in both statistical tests, considering kappa and accuracy measures.

In summary, the coevolutionary process performed by CIW-NN can be viewed as a strong improvement on the capabilities of the basic techniques. The individual benefits of each one are inherited by CIW-NN, obtaining the reduction capabilities of IS-CHC and SSMA and able to overcome statistically all the techniques considered. Consequently, these results show that CIW-NN is a suitable option to enhance the 1-NN rule in standard classification domains.

B. Comparison Between CIW-NN and Weighting Methods for k-NN-Based Classification

In this section, we perform a comparison between CIW-NN and several weighting methods for k-NN classification, ranging from classical approaches to more recent ones. We will check if CIW-NN is a competitive weighting method for the k-NN rule, in contrast with the existing techniques.

Table VI shows the average results obtained in the 30 small data sets of the general framework. Furthermore, we also report the average ranks computed by Friedman and Friedman aligned-ranks procedures. Fig. 8 shows the accuracy results in the test phase graphically, showing average results and standard deviations.

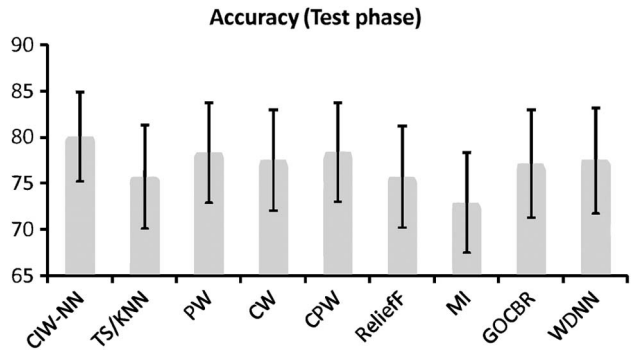


Fig. 8. Graphical comparison of accuracy in the test phase for CIW-NN and weighting methods.

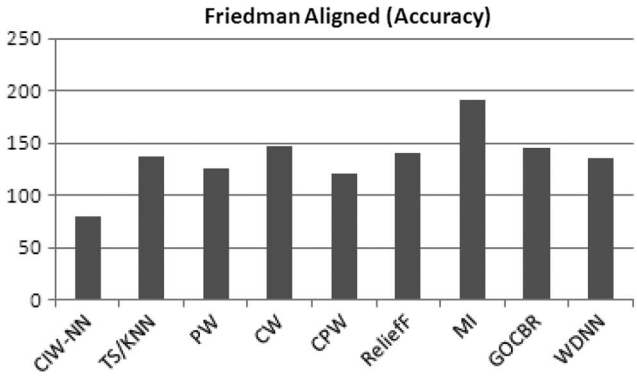


Fig. 9. Rankings computed by Friedman aligned-ranks procedure by using the accuracy measure.

The results obtained show that CIW-NN is the best performing algorithm in the test phase. This fact is reinforced by the average rankings obtained by the Friedman and Friedman aligned-ranks methods. Fig. 9 shows this comparison graphically for the Friedman aligned-ranks procedure with accuracy measure.

After computing the ranks, the Friedman and Friedman aligned-ranks procedures obtained p-values of 0.00005 and 0.00082 for accuracy and p-values of 0.04971 and 0.00090 for kappa, respectively. Thus, the two tests detected significant differences between the methods (final p-values computed by the *post hoc* methods are reported on the associated Internet site).

TABLE VII
AVERAGE RESULTS OBTAINED IN THE STUDY OF CIW-NN IN LARGE DOMAINS

	Accuracy		Kappa		Reduction	Time	Ranks (Acc.)		Ranks (Kap.)	
Method	Training	Test	Training	Test	Red.	Time (s.)	Fried.	F. Alig.	Fried.	F. Alig.
CIW-NN	74.56±0.54	75.67±2.19	0.6320±0.0139	0.6591±0.0239	0.8597	6135.42	2.62	19.62	3.25	23.87
IS-CHC	74.46±0.63	74.52±2.17	0.6467±0.0158	0.6210±0.0249	0.9934	1925.85	4.00	28.87	4.87	34.75
IS-SSMA	77.70±0.47	73.24±1.36	0.6765±0.0090	0.6025±0.0269	0.9780	1865.37	5.62	42.25	5.62	43.87
PW	74.42±0.58	72.60±1.28	0.6660±0.0121	0.6124±0.0301	-	16.08	5.50	43.62	4.75	39.75
CW	61.80±0.70	72.45±1.39	0.6219±0.0111	0.6036±0.0280	-	15.94	7.00	46.75	7.06	45.81
CPW	63.36±0.33	73.83±1.02	0.6722±0.0059	0.6173±0.0227	-	16.52	4.81	34.93	4.25	38.37
ReliefF	70.89±1.66	70.86±2.14	0.5581±0.0408	0.5578±0.0515	-	162.55	6.87	56.50	6.50	42.12
MI	69.81±0.68	69.49±1.56	0.5936±0.0059	0.5518±0.0304	-	5.57	6.25	42.75	6.87	52.75
WDNN	80.22±0.41	72.97±1.18	0.7308±0.0067	0.6149±0.0235	672.92	0.8489	4.93	39.43	4.12	37.25
1-NN	72.07±0.24	72.09±1.30	0.5988±0.0048	0.5998±0.0290	-	-	7.37	48.00	7.68	46.43

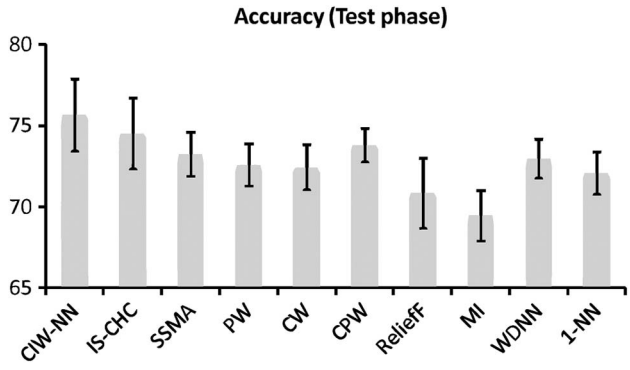


Fig. 10. Graphical comparison of accuracy in test phase in large domains.

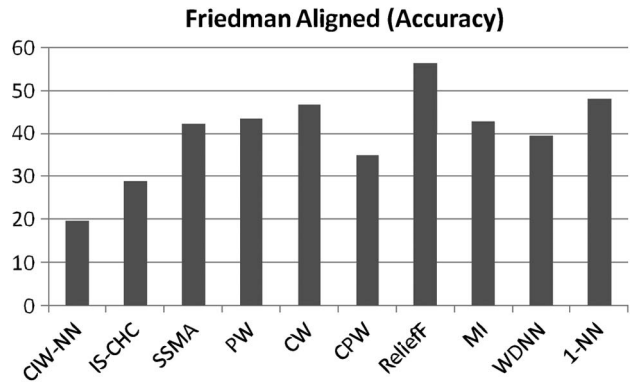


Fig. 11. Rankings of Friedman aligned-ranks test using accuracy in large domains.

From all the results shown, we can conclude the following.

1) CIW-NN offers the best accuracy and kappa results.
2) The average reduction rates achieved by CIW-NN are comparable with those achieved by WDNN.
3) CIW-NN achieves the best average rankings in the Friedman and Friedman aligned-ranks procedures, both in kappa and accuracy measures.

We can conclude this part of the study by stating that our approach is very competitive when compared with the rest of the methods considered. Its classification performance becomes a great advantage when compared with other techniques. Furthermore, with the application of CIW-NN, it is possible to obtain highly reduced training subsets for the 1-NN rule, comparable with the subsets achieved by WDNN (the only comparison method that is able to select weights and reduce the training set simultaneously).

C. Study of the Behavior of CIW-NN in Large-Sized Domains

In this paper, we will apply CIW-NN to the eight large data sets described in the general framework, with the aim of characterizing its behavior as the size of the problem increases. We have considered all the comparison methods except FW-SSGA, IW-SSGA, TS/KNN, and GOCBR, due to its high computational cost.

Table VII shows the average results obtained, highlighting CIW-NN as the best performing method in the test phase. This is also shown in Fig. 10. The rankings obtained by the Friedman aligned-ranks method are shown in Fig. 11.

In this comparison, only the Friedman test detected significant differences (p-values of 0.04790 and 0.04700 for accuracy and kappa, respectively, whereas the Friedman aligned-ranks p-values were 0.61974 and 0.63096). The final p-values computed by the *post hoc* methods are shown on the associated Internet site.

With these results, we can conclude the following.

1) CIW-NN has the best results in accuracy and kappa.
2) The reduction rates achieved by CIW-NN are comparable with those achieved by WDNN. However, they are lower than those of IS-CHC and SSMA.
3) CIW-NN achieves the best average rankings in the Friedman and Friedman aligned-ranks tests, with both measures.

All these results show us that CIW-NN is a very competitive algorithm in large domains. It still achieves better accuracy and kappa results than the rest of the techniques. Moreover, it also maintains a reasonable reduction rate; thus, its test classification phase will be faster than most of the remaining techniques.

D. Convergence Analysis

Often, the dynamics of CAs are hard to manage, since the constant changes in the global solution performed by the populations may provoke changes in their search space, thus changing the fitness value of their individuals [57]. However, CIW-NN's sequential scheme of evolution prevents these

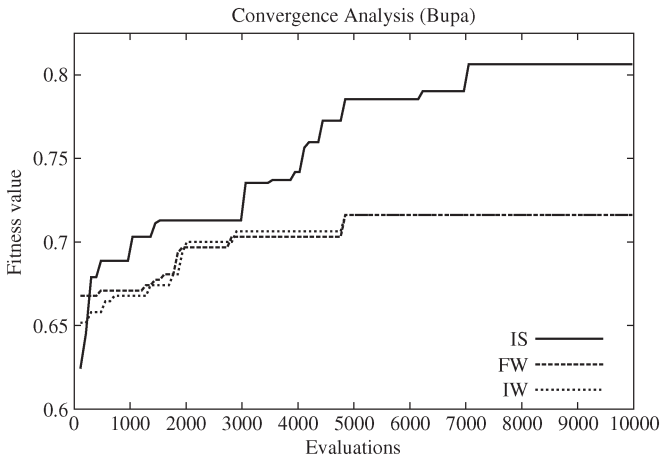


Fig. 12. Convergence analysis for CIW-NN in the Bupa problem. Critical points are found at 3000 and 4800 evaluations approximately, where the progress of a population allows the rest to escape from local maximums.

changes from decreasing the fitness value of the current best individual. In fact, this change may give an opportunity to the rest of the populations to escape from local optima, as the current *environment* (the parts of the solution already fixed by other populations) has been slightly changed. This is a key property of CAs.

Here, we show a representative example to illustrate this behavior. Fig. 12 shows the progress of the best chromosome of CIW-NN populations during the search.

In this example, the IS populations fall on a local optimum when roughly 1500 evaluations have been spent. However, FW and IW are still able to advance, thus modifying the current global solution and allowing the IS population to escape from the local optimum, when almost 3000 evaluations have been spent. This feedback also benefits the progress of the IW and FW populations. In our example, both suffer from stagnation when 3000 evaluations have been spent. However, further progress made by the IS populations allows them to escape from stagnation later (4800 evaluations). This allows them to improve the quality of their best solutions, to a point that they would not be able to reach by themselves.

VI. Conclusion

In this paper, we have presented CIW-NN, a novel evolutionary approach which integrates IS and two weighting schemes, i.e., FW and IW, with the aim of enhancing the results of the 1-NN classifier in supervised classification domains.

The application of a coevolutionary scheme has allowed us to integrate these techniques into a single method, by managing different EAs, particularly suited to their assigned tasks. Several mechanisms have been used to improve this cooperation, ranging from the use of specialized crossover operators (modified HUX and crossover multiple descendants) to the development of an epoch scheme designed to balance the intensity of the search in each of the populations.

We have performed a wide experimental study justifying the most important decisions taken in the designing process of CIW-NN. Moreover, we have shown that it is able to effectively improve the behavior of the 1-NN rule to a greater extent than a representative set of related evolutionary and nonevolutionary techniques. These results have been successfully contrasted by several nonparametric statistical procedures.

As future work, the employment of new search methods to support IS, FW, and IW processes may lead to promising results. For example, multiobjective methods for evolutionary IS or the development of new search methods for FW and IW with greater convergence capabilities (allowing, for example, the definition of a weight for each instance of the problem, instead of for each class) may improve the results of CIW-NN, achieving better performances in classification.

Acknowledgment

J. Derrac holds an FPU scholarship from the Spanish Ministry of Education.

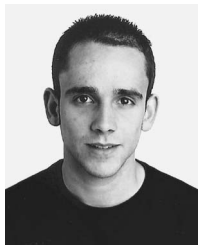
References

[1] E. Alpaydin, *Introduction to Machine Learning*, 2nd ed. Cambridge, MA: MIT Press, 2010.
[2] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, 2nd ed. Chichester, U.K.: Wiley, 2000.
[3] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference and Prediction*, 2nd ed. New York: Springer-Verlag, 2009.
[4] I. H. Witten, E. Frank, and M. A. Hall, *Data Mining: Practical Machine Learning Tools and Techniques*. San Mateo: Morgan Kaufmann, ser. Morgan Kaufmann Series in Data Management Systems, 3rd ed. 2011.
[5] D. W. Aha, Ed., *Lazy Learning*. New York: Springer-Verlag, 1997.
[6] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vol. IT-13, no. 1, pp. 21–27, 1967.
[7] X. Wu and V. Kumar, *The Top Ten Algorithms in Data Mining*, ser. Data Mining and Knowledge Discovery. London, U.K.: Chapman & Hall, 2009.
[8] Y. Chen, E. K. Garcia, M. R. Gupta, A. Rahimi, and L. Cazzanti, "Similarity-based classification: Concepts and algorithms," *J. Mach. Learn. Res.*, vol. 10, pp. 747–776, Dec. 2009.
[9] D. W. Aha, D. Kibler, and M. K. Albert, "Instance-based learning algorithms," *Mach. Learn.*, vol. 6, no. 1, pp. 37–66, Jan. 1991.
[10] C. G. Atkeson, A. W. Moore, and S. Schaal, "Locally weighted learning," *Artif. Intell. Rev.*, vol. 11, pp. 11–73, 1997.
[11] D. Wettschereck, D. W. Aha, and T. Mohri, "A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms," *Artif. Intell. Rev.*, vol. 11, pp. 273–314, 1997.
[12] D. R. Wilson and T. R. Martinez, "Reduction techniques for instance-based learning algorithms," *Mach. Learn.*, vol. 38, no. 3, pp. 257–286, 2000.
[13] I. Triguero, J. Derrac, S. García, and F. Herrera, "A taxonomy and experimental study on prototype generation for nearest neighbor classification," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 42, no. 1, pp. 86–100, Jan. 2012.
[14] B. Li, Y. Chen, and Y. Chen, "The nearest neighbor algorithm of local probability centers," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 38, no. 1, pp. 141–154, Feb. 2008.
[15] D. Pyle, *Data Preparation for Data Mining*. San Mateo, CA: Morgan Kaufmann, ser. The Morgan Kaufmann Series in Data Management Systems, 1999.
[16] H. Liu and H. Motoda, Eds., *Instance Selection and Construction for Data Mining*. New York: Springer-Verlag, 2001, ser. The Springer International Series in Engineering and Computer Science.
[17] M. Z. Jahromi, E. Parvinnia, and R. John, "A method of learning weighted similarity function to improve the performance of nearest neighbor," *Inf. Sci.*, vol. 179, no. 17, pp. 2964–2973, Aug. 2009.
[18] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*. New York: Springer-Verlag, ser. Natural Computing, 2003.
[19] A. A. Freitas, *Data Mining and Knowledge Discovery With Evolutionary Algorithms*. New York: Springer-Verlag, 2002.
[20] A. Ghosh and L. C. Jain, Eds., *Evolutionary Computation in Data Mining*. New York: Springer-Verlag, 2005.

[21] G. L. Pappa and A. A. Freitas, *Automating the Design of Data Mining Algorithms: An Evolutionary Computation Approach*. New York: Springer-Verlag, ser. Natural Computing, 2009.
[22] A. Fernandez, S. Garcia, J. Luengo, E. Bernado-Mansilla, and F. Herrera, "Genetics-based machine learning for rule induction: State of the art, taxonomy and comparative study," *IEEE Trans. Evol. Comput.*, vol. 14, no. 6, pp. 913–941, Dec. 2010.
[23] J. R. Cano, F. Herrera, and M. Lozano, "Using evolutionary algorithms as instance selection for data reduction in KDD: An experimental study," *IEEE Trans. Evol. Comput.*, vol. 7, no. 6, pp. 561–575, Dec. 2003.
[24] L. I. Kuncheva, "Editing for the k-nearest neighbors rule by a genetic algorithm," *Pattern Recognit. Lett.*, vol. 16, no. 6, pp. 809–814, Aug. 1995.
[25] S.-Y. Ho, C.-C. Liu, and S. Liu, "Design of an optimal nearest neighbor classifier using an intelligent genetic algorithm," *Pattern Recognit. Lett.*, vol. 23, no. 13, pp. 1495–1503, Nov. 2002.
[26] S. García, J. R. Cano, and F. Herrera, "A memetic algorithm for evolutionary prototype selection: A scaling up approach," *Pattern Recognit.*, vol. 41, no. 8, pp. 2693–2709, Aug. 2008.
[27] S. García and F. Herrera, "Evolutionary undersampling for classification with imbalanced datasets: Proposals and taxonomy," *Evol. Comput.*, vol. 17, no. 3, pp. 275–306, Nov. 2009.
[28] A. Cervantes, I. Galván, and P. Isasi, "AMPSO: A new particle swarm method for nearest neighborhood classification," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 5, pp. 1082–1091, Oct. 2009.
[29] M. A. Potter and K. A. D. Jong, "Cooperative coevolution: An architecture for evolving coadapted subcomponents," *Evol. Comput.*, vol. 8, no. 1, pp. 1–29, 2000.
[30] M. Li and Z. Wang, "A hybrid coevolutionary algorithm for designing fuzzy classifiers," *Inf. Sci.*, vol. 179, no. 12, pp. 1970–1983, May 2009.
[31] P. Gançarski, A. Blansché, and A. Wania, "Comparison between two coevolutionary feature weighting algorithms in clustering," *Pattern Recognit.*, vol. 41, no. 3, pp. 983–994, Mar. 2008.
[32] Z. Yang, K. Tang, and X. Yao, "Large scale evolutionary optimization using cooperative coevolution," *Inf. Sci.*, vol. 178, no. 15, pp. 2985–2999, Aug. 2008.
[33] Y. Wang and A. Nakao, "On cooperative and efficient overlay network evolution based on a group selection pattern," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 40, no. 3, pp. 656–667, Jun. 2010.
[34] N. García-Pedrajas, C. Hervás-Martínez, and D. Ortiz-Boyer, "Cooperative coevolution of artificial neural network ensembles for pattern classification," *IEEE Trans. Evol. Comput.*, vol. 9, no. 3, pp. 271–302, Jun. 2005.
[35] N. García-Pedrajas, J. A. R. del Castillo, and D. Ortiz-Boyer, "A cooperative coevolutionary algorithm for instance selection for instance-based learning," *Mach. Learn.*, vol. 78, no. 3, pp. 381–420, Mar. 2010.
[36] J. Derrac, S. García, and F. Herrera, "IFS-CoCo: Instance and feature selection based on cooperative coevolution with nearest neighbor rule," *Pattern Recognit.* , vol. 43, no. 6, pp. 2082–2105, Jun. 2010.
[37] A. M. Sánchez, M. Lozano, P. Villar, and F. Herrera, "Hybrid crossover operators with multiple descendents for real-coded genetic algorithms: Combining neighborhood-based crossover operators," *Int. J. Intell. Syst.*, vol. 24, no. 5, pp. 540–567, May 2009.
[38] C. Rosin and R. Belew, "New methods for competitive coevolution," *Evol. Comput.*, vol. 15, no. 1, pp. 1–29, 1997.
[39] C.-K. Goh and K. C. Tan, "A competitive–cooperative coevolutionary paradigm for dynamic multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 1, pp. 103–127, Feb. 2009.
[40] R. P. Wiegand and J. Sarma, "Spatial embedding and loss of gradient in cooperative coevolutionary algorithms," in *Proc. 8th Int. Conf.—PPSN*, Birmingham, U.K., Sep. 18–22, 2004, vol. 3242, pp. 912–921.
[41] X. Li and X. Yao, "Tackling high dimensional nonseparable optimization problems by cooperatively coevolving particle swarms," in *Proc. IEEE CEC*, 2009, vol. 9, pp. 1546–1553.
[42] E. Popovici and K. A. D. Jong, "Sequential versus parallel cooperative coevolutionary algorithms for optimization," in *Proc. IEEE CEC*, Vancouver, BC, Canada, 2006, vol. 3242, pp. 1610–1617.
[43] E. Popovici and K. A. D. Jong, "The effects of interaction frequency on the optimization performance of cooperative coevolution," in *Proc. GECCO*, Seattle, WA, Jul. 8–12, 2006, pp. 353–360.
[44] L. Panait, S. Luke, and J. F. Harrison, "Archive-based cooperative coevolutionary algorithms," in *Proc. GECCO*, Seattle, WA, Jul. 8–12, 2006, pp. 345–352.
[45] S. García, J. Derrac, J. R. Cano, and F. Herrera, "Prototype selection for nearest neighbor classification: Taxonomy and empirical study," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 3, pp. 417–435, Mar. 2012.
[46] F. Fernández and P. Isasi, "Local feature weighting in nearest prototype classification," *IEEE Trans. Neural Netw.*, vol. 19, no. 1, pp. 40–53, Jan. 2008.
[47] I. Kononenko, "Estimating attributes: Analysis and extensions of RELIEF," in *Proc. Eur. Conf. Mach. Learn.*, Catania, Italy, 1994, pp. 171–182.
[48] R. Paredes and E. Vidal, "Learning weighted metrics to minimize nearest-neighbor classification error," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 7, pp. 1100–1110, Jul. 2006.
[49] L. J. Eshelman, "The CHC adaptative search algorithm: How to have safe search when engaging in nontraditional genetic recombination," in *Foundations of Genetic Algorithms*, G. J. E. Rawlins, Ed. San Mateo, CA: Morgan Kaufmann, 1991, pp. 265–283.
[50] J. D. Kelly and L. Davis, "A hybrid genetic algorithm for classification," in *Proc. 12th IJCAI*, Sydney, Australia, 1991, vol. 2, pp. 645–650.
[51] H. Ahn and K. Kim, "Bankruptcy prediction modeling with hybrid case-based reasoning and genetic algorithms approach," *Appl. Soft Comput.*, vol. 9, no. 2, pp. 599–607, Mar. 2009.
[52] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs (2nd, Ext. ed.)*. New York: Springer-Verlag, 1994.
[53] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, and F. Herrera, "Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework," *J. Multiple-Valued Logic Soft Comput.*, vol. 17, no. 2/3, pp. 255–287, 2011.
[54] M. A. Tahir, A. Bouridane, and F. Kurugollu, "Simultaneous feature selection and feature weighting using hybrid tabu search/ k-nearest neighbor classifier," *Pattern Recognit. Lett.*, vol. 28, no. 4, pp. 438–446, Mar. 2007.
[55] J. Cohen, "A coefficient of agreement for nominal scales," *Edu. Psychol. Meas.*, vol. 20, no. 1, pp. 37–46, Apr. 1960.
[56] S. García, A. Fernández, J. Luengo, and F. Herrera, "Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," *Inf. Sci.*, vol. 180, no. 10, pp. 2044–2064, May 2010.
[57] L. Panait, "Theoretical convergence guarantees for cooperative coevolutionary algorithms," *Evol. Comput.*, vol. 18, no. 4, pp. 581–615, Nov. 2010.

Joaquín Derrac received the M.Sc. degree in computer science from the University of Granada (UGR), Granada, Spain, in 2008, where he is currently working toward the Ph.D. degree in the Department of Computer Science and Artificial Intelligence.

His research interests include data mining, data reduction, statistical inference, and evolutionary algorithms.



Isaac Triguero received the M.Sc. degree in computer science from the University of Granada (UGR), Granada, Spain, in 2009, where he is currently working toward the Ph.D. degree in the Department of Computer Science and Artificial Intelligence.

His research interests include data mining, semi-supervised learning, data reduction, and evolutionary algorithms.

Salvador García received the M.Sc. and Ph.D. degrees in computer science from the University of Granada (UGR), Granada, Spain, in 2004 and 2008, respectively.

He is currently an Assistant Professor with the Department of Computer Science, University of Jaén, Jaén, Spain. He has more than 25 papers published in international journals. He has coedited two special issues of international journals on different data mining topics. His research interests include data mining, data reduction, data complexity, imbalanced learning, semisupervised learning, statistical inference, and evolutionary algorithms.

Francisco Herrera (M'10) received the M.Sc. and Ph.D. degrees in mathematics from the University of Granada (UGR), Granada, Spain, in 1988 and 1991, respectively.

He is currently a Professor with the Department of Computer Science and Artificial Intelligence, UGR. He currently acts as the Editor-in-Chief of the international journal "Progress in Artificial Intelligence" (Springer) and serves as an Area Editor of the journal *Soft Computing* (area of evolutionary and bioinspired algorithms) and *International Journal of Computational Intelligence Systems* (area of information systems). He acts as an Associate Editor of the journals *Information Sciences*, *Advances in Fuzzy Systems*, and *International Journal of Applied Metaheuristics Computing*, and he serves as a member of several journal editorial boards, such as *Fuzzy Sets and Systems*, *Applied Intelligence*, *Knowledge and Information Systems*, *Information Fusion*, *Evolutionary Intelligence*, *International Journal of Hybrid Intelligent Systems*, *Memetic Computing*, and *Swarm and Evolutionary Computation*. He is a coauthor of the book "Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases" (World Scientific, 2001). He has more than 200 papers published in international journals. His current research interests include computing with words and decision making, data mining, bibliometrics, data preparation, instance selection, fuzzy-rule-based systems, genetic fuzzy systems, knowledge extraction based on evolutionary algorithms, memetic algorithms, and genetic algorithms.

Dr. Herrera was the recipient of the following honors and awards: European Coordinating Committee for Artificial Intelligence (ECCAI) Fellow 2009, 2010 Spanish National Award on Computer Science ARITMEL to the "Spanish Engineer on Computer Science," and International Cajastur "Mamdani" Prize for Soft Computing (Fourth Edition, 2010). He acts as an Associate Editor of IEEE TRANSACTIONS ON FUZZY SYSTEMS.