Pattern Recognition 43 (2010) 2082–2105

Contents lists available at ScienceDirect

Pattern Recognition

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

IFS-CoCo: Instance and feature selection based on cooperative coevolution with nearest neighbor rule ☆

Joaquín Derrac [a,*,1], Salvador García [b], Francisco Herrera [a]

[a] Department of Computer Science and Artificial Intelligence, CITIC-UGR (Research Center on Information and Communications Technology), University of Granada, 18071 Granada, Spain

[b] Department of Computer Science. University of Jaén, 23071 Jaén, Spain

ARTICLE INFO

Article history:
Received 25 June 2009
Received in revised form
6 December 2009
Accepted 17 December 2009

Keywords:
Evolutionary algorithms
Feature selection
Instance selection
Cooperative coevolution
Nearest neighbor

ABSTRACT

Feature and instance selection are two effective data reduction processes which can be applied to classification tasks obtaining promising results. Although both processes are defined separately, it is possible to apply them simultaneously.

This paper proposes an evolutionary model to perform feature and instance selection in nearest neighbor classification. It is based on cooperative coevolution, which has been applied to many computational problems with great success.

The proposed approach is compared with a wide range of evolutionary feature and instance selection methods for classification. The results contrasted through non-parametric statistical tests show that our model outperforms previously proposed evolutionary approaches for performing data reduction processes in combination with the nearest neighbor rule.

© 2009 Elsevier Ltd. All rights reserved.

1. Introduction

The designing of classifiers can be considered one of the main processes inside the data mining field. Due to the large amount of data generated in many research areas, ranging from human genome sequenciation projects to development of new technical prototypes in industry, the use of machine learning algorithms has become a challenging task [1,2].

The employment of data reduction [3] techniques in the first phases of the construction of classifiers is a necessity in most data mining applications nowadays. The main objectives of these techniques are to increase the efficiency of the classification process (by removing redundant instances and features or discretizing variables) and to reduce the classification error rate (by removing noisy instances and features). Although data reduction techniques were originally designed to work with standard data, it is not difficult to find applications of data reduction in other fields, e.g. dealing with multimedia data [4] or graphics [5]. Data reduction is also used to optimize dissimilarity-based classification [6], to obtain high quality rules in high-dimensional subgroup discovery problems [7] and to enhancing the data quality based on complexity measures as the computation of volume-based inter-class overlap measures [8].

One of the most well known classifiers is the *k*-Nearest Neighbors classifier (*k*-NN) [9]. It has been applied to many classification problems [10]. It is a non-parametric classifier which does not build a model in its training phase. Instead of using a model, it is based on the instances contained in the training set. Thus, the effectiveness of the classification process relies 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 grows, regarding both the number of examples in the data set and the number of attributes which will be used in the computation of its similarity functions (distances).

For this contribution, two well-known techniques of data reduction will be employed: instance selection (IS) [11] and feature selection (FS) [12]. The objective of IS is to select the most appropriate subset of instances (prototypes) from the initial data, trying simultaneously to increase the accuracy of the classification process and decrease the amount of data employed in it. FS works in a similar way, selecting the most appropriate subset of features to describe the data. Both are really effective not only in reducing the size of the initial data set, but also in filtrating and cleaning noisy data. In the field of machine learning, we can find interesting approaches [11,12], some of them trying to enhance the results obtained by the *k*-NN classifier [13].

On the other hand, the research done in evolutionary computation (EC) [14] has contributed numerous techniques inspired by natural evolution, which are able to manage search

☆This work was supported by Project TIN2008-06681-C06-01.

* Corresponding author. 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).

[1] The author holds a research scholarship from the University of Granada.

0031-3203/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2009.12.012

problems like IS [13,15,16] and FS [17–19]. Furthermore, Evolutionary Algorithms (EAs) have been successfully used in data mining problems, showing that they are a very useful tool to perform this task [20–22].

A more specialized approach can be found in coevolution [23], a specialized trend of EAs. It works by managing two or more populations (also called species) simultaneously, allowing interactions among its individuals. This approach allows splitting the problem into different parts, employing a population to handle each one separately, but joining its individuals to evaluate the solutions obtained. Recently, the coevolution model has shown some interesting characteristics [24], being successfully applied to different problems [25,26].

Our proposal defines a cooperative coevolution model to tackle the IS and FS problems. In the instance and feature selection based on the cooperative coevolution (IFS-CoCo) model, both processes are applied simultaneously to the initial data set, aiming to obtain a suitable training set to perform the classification process.

IFS-CoCo is composed of three populations. The individuals of each one define a different type of baseline classifier, depending on each population's characteristics. Thus, each population is focused on performing a basic data reduction task: The first population performs an IS process, the second population performs a FS process, and the third population performs simultaneously both IS and FS processes. With the employment of coevolution, this approach is intended to improve the results of data reduction techniques when applied to classification tasks.

In this work, IFS-CoCo will be fully described, from its theoretical background to the details of its implementation. Moreover, a wide range of classification problems will be employed to perform a comparison between IFS-CoCo and other models, in order to highlight the benefits of the use of coevolution. We will employ a Wilcoxon signed-ranks test [27] to contrast the results obtained.

The rest of the paper is organized as follows: Section 2 summarizes the existing work in the related areas. Section 3 describes the cooperative coevolutive model proposed. Section 4 deals with the experimental framework employed. Section 5 presents the analysis of results. Section 6 shows the conclusions arrived at. Finally, two appendices are provided to extend the details of the experimental study performed. Appendix A describes the main characteristics of the comparison algorithms employed. Appendix B shows the complete results obtained.

2. Background: data reduction and coevolutionary algorithms

This section discusses the main topics in the background in which our contribution is based. Section 2.1 describes in depth IS and FS as data reduction techniques. Section 2.2 shows some examples of how EAs can be applied to data reduction problems. Finally, Section 2.3 highlights the main characteristics of coevolutionary algorithms.

2.1. Data reduction techniques

Two well-known data reduction techniques are going to be reviewed in this subsection: IS and FS. In addition, an analysis of simultaneous instance and feature selection (IFS) will be provided. This will cover all the background needed to understand the data reduction processes performed by IFS-CoCo.

2.1.1. Instance selection

IS is one of the main data reduction techniques. In IS, the 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 quality as the initial data set [11]. By minimizing the data set size, it is possible to reduce the space complexity and decrease the computational cost of the data mining algorithms that will be applied later, improving their generalization capabilities through the elimination of noise.

More specifically, IS can be defined as follows: Let X_p be an instance where $X_p = (X_{p1}, X_{p2}, \ldots, X_{pm}, X_{pc})$, with X_p belonging to a class c given by X_{pc}, and a m-dimensional space in which X_{pi} is the value of the i-th feature of the p-th sample. Then, let us assume that there is a training set TR which consists of N instances X_p and a test set TS composed of T instances X_p. Let $S \subseteq TR$ be the subset of selected samples that resulted from the execution of a IS algorithm, then we classify a new pattern T from TS by a data mining algorithm acting over the instances of S.

IS methods can be divided into two categories: prototype selection (PS) methods and training set selection (TSS) methods. PS methods [28] are IS methods which expect to find training sets offering the best classification accuracy and reduction rates by using instance based classifiers which consider a certain similarity or distance measure (e.g., k-NN). On the other hand, TSS methods are known as the application of IS methods over the training set to build any predictive model (e.g. decision trees, neural networks [29,30]).

In this work, we will focus our attention on PS, because we will employ the nearest neighbor rule as the baseline rule to perform the classification process. More concretely, we will employ the 1-NN rule. Wilson and Martinez, in [31], suggest that the determination of the k value in the k-NN classifier may depend on the proposal of the IS algorithm. Setting $k > 1$ decreases the sensitivity of the algorithm to noise and tends to smooth the decision boundaries. In some IS algorithms, a value $k > 1$ may be convenient, when the interest lies in protecting the classification task of noisy instances. Therefore, Wilson et al states that it may be appropriate to find a value of k to use during the reduction process, and then redetermine the best value of k in the classification task. For this contribution, we have employed the value $k = 1$, given that EAs need to have the greatest possible sensitivity to noise during the reduction process. In this manner, an evolutionary IS algorithm could better detect the noisy instances and the redundant ones in order to find a subset of instances adapted to the simplest method of nearest neighbors.

In the data mining field many approaches of PS have been developed, ranging from classical approaches such as CNN [32] or ENN [33] to recent approaches such as SSMA [15], HMNEI [34] or PSC [35]. A wide number of reviews of PS methods can be found in the literature [36–38,31].

2.1.2. Feature selection

FS is another of the main data reduction techniques. In FS, the goal is to select the most appropriate subset of features from the initial data set. It aims to eliminate irrelevant and/or redundant features to obtain a simple and accurate classification system [12].

FS can be defined as follows: Let X_p be an instance where $X_p = (X_{p1}, X_{p2}, \ldots, X_{pm}, X_{pc})$, with X_p belonging to a class c given by X_{pc}, and an m-dimensional space in which X_{pi} is the value of the i-th feature of the p-th sample. Then let us assume that there is a training set TR whose instances X_p are defined by M features, and a test set TS. Let $P \subseteq M$ be the subset of selected features that resulted from the execution of a FS algorithm, then we classify a new pattern from TS by a data mining algorithm acting over TR, employing for reference only the features selected in P.

There are three main categories in which FS methods can be classified:

- *Wrapper methods*, where the selection criterion is dependent on the learning algorithm, being a part of the fitness function [39].
- *Filtering methods*, where the selection criterion is independent of the learning algorithm (separability measures are employed to guide the selection) [40].
- *Embedded methods*, where the search for an optimal subset of features is built into the classifier construction [41].

As with IS methods, a great number of FS methods have been developed recently. Two of the most well known classical algorithms are forward sequential and backward sequential selection [42], which begin with a feature subset and sequentially add or remove features until the finalization of the algorithm.

Despite the popularity of sequential methods, other approaches can be found in the literature [43]. Some of them are based on heuristics [44], showing a proof of heuristics and metaheuristics can be very useful in the task of selecting the most appropriate subset of features to be used in a classification algorithm. More complex approaches have been developed, based on fuzzy entropy measures [45]. Some complete surveys, analyzing both classical and advanced approaches to FS, can be found in the literature [40,46,41].

2.1.3. Instance and feature selection

Instead of approaching IS or FS problems separately, some research efforts have been applied to the study of the dual IS and FS problem, which we will denote Instance and Feature Selection (IFS). There is no inconvenience in tackling both problems simultaneously because features and instances can be selected in an independent way: the classification accuracy of the classification process is the only part of the problem affected, which will be determined by the selected data.

Some proposals for IFS can be found in the literature; a first approach is proposed in [47], where a Genetic Algorithm (GA) is employed to simultaneously select suitable instances and features for a reference set of a k-NN classifier. This approach was improved in [48], where, in addition, their proposal was also employed to improve the performance of neural networks in classification.

Another recent proposal can be found in [49], where a simulated annealing method [50] is applied to perform alternately IS and FS on each step of the search. More complex approaches mixing Feature Weighting, IS and FS have been developed recently [51,10].

2.2. Evolutionary algorithms on data reduction

Recently, the employment of EAs in data reduction problems has become common in the machine learning field. This subsection will review some interesting examples.

In [13], a complete study of the use of EAs in IS is made, highlighting four EAs to complete this task: CHC Adaptive Search Algorithm (CHC) [52], Steady-State Genetic Algorithm (SSGA), Generational Genetic Algorithm (GGA) and population-based incremental learning (PBIL). They concluded that EAs outperform classical algorithms both in reduction rates and classification accuracy. They also concluded that CHC is an appropriate EA to carry out this task, according to the algorithms they compared. Other proposals can be found in [15,53,54,16,55].

Most of the EAs approaches in FS are based on GAs, using both filter and wrapper approaches [56–58,18,59–62]. A remarkable proposal is [19], where the CHC algorithm shows good results when applied to FS problems. Another interesting proposal is [17], where an estimation of distribution algorithm based on Bayesian Networks is presented.

It is possible to find applications of simultaneous IS and FS to EAs. Both [47] and [48] propose a GA to perform the editing of the instance set and selection of the feature set. Ho et al [63] presented IGA, an intelligent GA designed to tackle both IS and FS problems simultaneously, by the introduction of a special orthogonal cross operator. More recently, a hybrid GA (HGA) [64] has been developed by merging local search optimization techniques with the genetic component itself. HGA performs its search in two phases, firstly by using a basic GA based on the restricted tournament selection (RTS) scheme, and secondly by employing some different processes of local searches to help the GA to converge.

2.3. Coevolutionary algorithms

A coevolutionary algorithm (CA) is an EA which is able to manage two or more populations simultaneously. Coevolution, the field in which CAs can be classified, can be defined as the coexistence of some interacting populations, evolving simultaneously. In this manner, evolutionary biologist Price [65] defined coevolution as *reciprocally induced evolutionary change between two or more species or populations*. A wider discussion about the meaning of coevolution in the field of EC can be found in the dissertation thesis of Wiegand [66].

The most important characteristic of coevolution is the possibility of splitting a given problem into different parts, employing a population to handle each one separately. This allows the algorithm 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 coevolution model can show interesting benefits in its application.

Therefore, the interaction between individuals of different populations is key to the success of coevolution techniques. In the literature, coevolution is often divided into two classes, regarding the type of interaction employed:

Cooperative coevolution (*CoCo*): In this trend, each population evolves individuals representing a component of the final solution. Thus, a full candidate solution is obtained by joining an individual chosen from each population. In this way, increases in a collaborative fitness value are shared among individuals of all the populations of the algorithm [23].

Competitive coevolution (*ComCo*): 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 [67].

Coevolution is a research field which has started to grow recently. With respect to the architecture of its models, some interesting topics can be remarked upon:

- Some research efforts have been applied to tackle the question about how to select the members of each population that will be used to evaluate the fitness function. One way is to evaluate an individual against every single collaborator in the other population [68]. Although it would be a better way to select the collaborators, it would consume a very high number of evaluations in the computation of the fitness function. 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 [69].

- One main problem which CoCo (and Coevolution, in general) must face is known as *the loss of gradient problem* in which one population comes to severely dominate the others, creating a situation where the other populations have insufficient information from which to learn, due to the high degree of domination present. This problem has been addressed by several authors [70].
- Another question to solve is to define how the algorithm should manage its populations. The most common answers are to manage them by using either a sequential scheme or a parallel scheme. Several studies have been done comparing both approaches [71].
- The assignation of fitness to each individual is also an open question. This feature, also called *Collaboration Credit Assignment* is the rule which defines how a fitness value of an individual will be updated when it will be used two or more times as a part of a complete solution. Although the simplest solution is to use just the last given value, some different schemes have been developed, e.g., *minimum*, *maximum and average* [72]. Also, more complex relationships have been developed, e.g., based on game theory [73].

All these advances have been proposed with a main idea in mind: coevolution is able to beat the well known No-Free-Lunch (NFL) barrier present on most of the function optimization techniques [74]. That means that it is possible to design CAs which perform better than others EAs, when averaged over all interaction functions, with respect to some measure of performance [24]. This theoretical result has been studied in depth, and as a result some proposals of NFL frameworks for Coevolution have been developed [75].

3. A cooperative coevolutionary algorithm for instance and feature selection: IFS-CoCo

The main features of IFS-CoCo will be presented in this section, as well as all the details needed to perform its implementation, along the next four subsections. Section 3.1 deals with the description of the populations and the chromosome representation. Section 3.2 defines the fitness function employed. Section 3.3 presents the baseline EA on which our model is based, the CHC algorithm. Finally, Section 3.4 describes the main coevolutionary process performed by IFS-CoCo.

3.1. Populations and chromosome representation

As mentioned in the first section of this work, IFS-CoCo manages three populations. The chromosomes of each one define a different type of baseline classifier, thus each population is focused on performing a basic data reduction task:

- The first population performs an IS process.
- The second population performs a FS process.
- The third population performs an IFS process.

From now on, they will be referred to as IS population, FS population and IFS population, respectively. Fig. 1 shows a basic representation of the scheme of the populations of IFS-CoCo.

All populations share the same basic chromosome definition. Let us assume a data set with N instances and M attributes. Each chromosome consists of a determinate number of genes, which represents either an instance or a feature. A binary representation is used, thus each gene has two possible states: 1, if the corresponding feature/instance is included in the data set represented by the chromosome, or 0 if not.

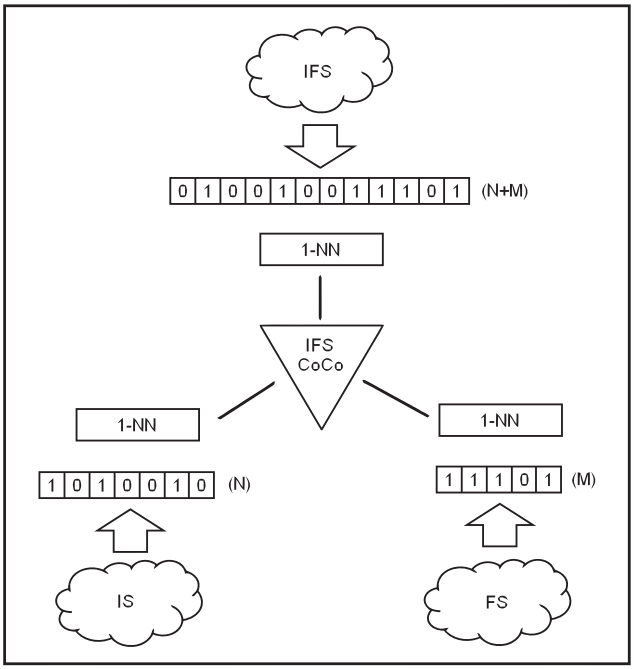

Fig. 1. Population scheme of IFS-CoCo.

The concrete representation and size of the chromosome depends on the population to which it belongs:

- IS population: Each gene represents an instance (chromosome size: N).
- FS population: Each gene represents a feature (chromosome size: M).
- IFS population: The first N genes of the chromosome represent instances. Remaining genes represent features (chromosome size: $N+M$).

By using this representation scheme, all chromosomes will define a subset of the original data set, with everyone focused on a concrete data reduction task. Regarding to the classification task, each chromosome symbolizes a reduced subset, which will be employed as a training set by the 1-NN classifier [9].

3.2. Fitness function

IFS-CoCo uses a fitness function focused on two objectives: Maximize the accuracy rate of the multiclassifier defined by the combination of the three populations and maximize the reduction rates over instances and features.

Three chromosomes are needed to compute the fitness function (one of each population, defining the three basic classifiers). Each chromosome will get a fitness value, depending on the accuracy rate obtained by the multiclassifier defined by the joint of the three chromosomes, and also depending on the reduction rate obtained by the data set coded by its phenotype.

To obtain the accuracy rate of the combination of three chromosomes it is necessary to build a multiclassifier. This task can be accomplished by building the three basic classifiers defined by the chromosomes (an IS classifier, an FS classifier

and an IFS classifier). Then, the three outputs must be joined into one, by using a majority voting function.

Each output value is obtained as the majority output of the three basic classifiers. If no majority vote can be obtained, then the output of the currently best classifier is preferred. The currently best classifier is the one that belongs to the population that achieved better overall results in the previous generation.

When the majority voting process has finished, the resulting class can be regarded as the output of the multiclassifier. At this point, the accuracy of the classifier, *classRate*, can be computed.

$$classRate(G,H,I) = \frac{\#Instances\ classified\ correctly}{N} \quad (1)$$

where G is a chromosome from the IS population, H is a chromosome of the FS population, I is a chromosome of the IFS population, and N is the number of instances in the training set. This result is assigned as the *classRate* of G, H and I.

$$\begin{aligned} classRate(G) &= classRate(G,H,I) \\ classRate(H) &= classRate(G,H,I) \\ classRate(I) &= classRate(G,H,I) \end{aligned} \quad (2)$$

On the other hand, the reduction rates can be computed from any chromosome. For a given chromosome J, two reduction rates are defined:

- *ReductionIS*, which symbolizes the reduction rate obtained regarding the instances of the data set:

$$ReductionIS(J) = 1.0 - \frac{\#Instances\ Selected}{N} \quad (3)$$

  Where *#Instances Selected* is the number of genes set to 1 if J belongs to the IS population, the number of genes set to 1 in the first N genes if J belongs to the IFS population, or N if J belongs to the FS population.
- *ReductionFS*, which symbolizes the reduction rate obtained regarding the features of the data set:

$$ReductionFS(J) = 1.0 - \frac{\#Features\ Selected}{M} \quad (4)$$

  Where *#Features Selected* is the number of genes set to 1 if J belongs to the FS population, the number of genes set to 1 in the last M genes if J belongs to the IFS population, or M if J belongs to the IS population.

At this point, assuming that a chromosome J has already defined its *classRate*, *ReductionIS* and *ReductionFS* values, its fitness value can be computed. The fitness function must be able to give any chromosome a suitable value which adequately represent those values.

A first approach (Eq. (5)) can be found in [13], where the *classRate* and *Reduction* (the reduction rate achieved over the selected instances) values are employed to define a suitable fitness function for the IS problem, employing an α real-valued weighting factor:

$$Fitness(J) = \alpha \cdot clasRate(J) + (1-\alpha) \cdot Reduction(J) \quad (5)$$

However, for this model we must define an expression composed by *classRate* and the two reduction rates, *ReductionIS* and *ReductionFS*. To obtain it, we define for IFS-CoCo the following fitness function:

$$Fitness(J) = \alpha \cdot \beta \cdot clasRate(J) + (1-\alpha) \cdot ReductionIS(J) + (1-\beta) \cdot ReductionFS(J) \quad (6)$$

where α and β are real-valued weighting factors valued in the interval [0,1].

In [13], it is suggested to employ a value of 0.5 for the α parameter (for Eq. (6)). In our approach, this value should be increased a bit, due to the influence of the simultaneous use of *ReductionIS* and *ReductionFS* components (and the β parameter) to compute the fitness value.

On the other hand, the value of the second parameter, β, should be adjusted carefully. This value has to be near 1.0, to avoid an excessive deletion of features in the solutions obtained. In the k-NN classifier, the removal of a feature can influence too much the subsequent classification process. Thus, a high pressure towards obtaining high reduction rates over the subset of selected features may degrade significantly the accuracy of the classifier. Consequently, the value of the β parameter should not be set very far from 1.0 (but lesser to it, to allow our model to select smaller subsets when comparing two solutions with a similar *classRate* associated).

3.3. CHC algorithm

CHC [52] is a binary-coded GA which involves the combination of a selection strategy with a very high selective pressure, and several components inducing a strong diversity. Due to these characteristics, CHC has become a robust EA, which should often offer promising results in several search problems.

We have selected CHC as a baseline EA for our model because it has been widely studied, being now a well-known algorithm on evolutionary computation. Furthermore, previous studies like [13] support the fact that it can perform well on data reduction problems.

The four main components of the algorithm are shown as follows:

- *An elitist selection*: The members of the current population are merged with the offspring population obtained from it and the best individuals are selected to compose the new population. In cases where a parent and an offspring have the same fitness value, the former is preferred to the latter.
- *A highly disruptive crossover*: HUX, which crosses over exactly half of the non-matching alleles, where the bits to be exchanged are chosen at random without replacement. This way, it guarantees that the two offspring are always at the maximum Hamming distance from their two parents, thus encouraging the introduction of a high diversity in the new population and lessening the risk of premature convergence.
- *An incest prevention mechanism*: During the reproduction step, each member of the parent (current) population is randomly chosen without replacement and paired for mating. However, not all these couples are allowed to cross over. Before mating, the Hamming distance between the potential parents is calculated and if half this distance does not exceed a difference threshold d, they are not mated. The aforementioned threshold is usually initialized to $L/4$ (with L being the chromosome length). If no offspring is obtained in one generation, the difference threshold is decremented by one.
  The effect of this mechanism is that only the more diverse potential parents are mated, but the diversity required by the difference threshold automatically decreases as the population naturally converges.
- *A restart process*: replacing the GA mutation, which is only applied when the population has converged. The difference threshold is considered to measure the stagnation of the search, which happens when it has dropped to zero and several generations have been run without introducing any new

individual into the population. Then, the population is reinitialized by considering the best individual as the first chromosome of the new population and generating the remaining chromosomes by randomly flipping a percentage (usually 35%) of their bits.

Algorithm 1 shows a basic pseudocode of CHC.

Algorithm 1. CHC algorithm basic structure

```
    Input: population
1   Initialization(population);
2   d = L/4;
3   Evaluate(population);
4   while termination condition not satisfied do
5     candidates=SelectParents(population);
6     offSpring=CrossParents(candidates);
7     Evaluate(offspring);
8     SelectNewPopulation(population, offspring);
9     if Population not changed then
10      d = d-1;
11    end
12    if d < 0 then
13      Restart(population);
14      Initialize(d);
15    end
16  end
    Output: Best(population)
```

To increase the speed of the data reduction process performed by IFS-CoCo, we have modified the definition of the HUX-cross operator. In this manner, when a gene representing an instance is going to be set from 0 to 1 by the crossing procedure, it is only set to one with a defined probability (*prob0to1* parameter). No modifications are applied to changes from 1 to 0, or to genes representing features.

For example, if one chromosome, 1100000000, and another chromosome, 1111111111, defining an IS classifier, are crossed by the HUX standard operator, the offspring will be 1111110000 and 1100001111. On the same scenery, an execution of our HUX modified operator, with a probability of change $prob0to1 = 0.5$, would give as the output the offspring 1101100000 and 1100001111. Fig. 2 shows this example graphically.

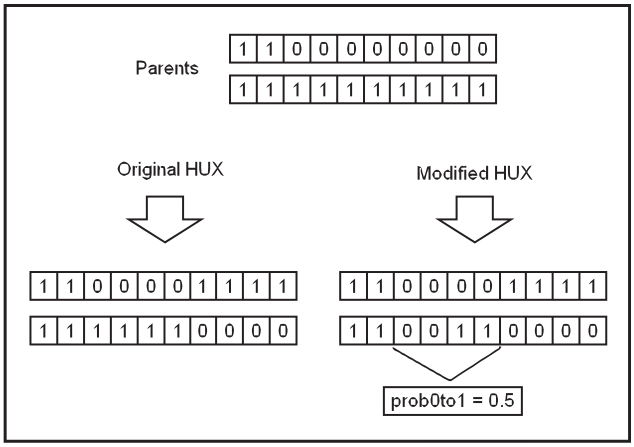

Fig. 2. The modified HUX crossing operator.

With this modification, the HUX crossing operator will help to speed up the reduction process.

3.4. Coevolutionary process

This subsection describes the coevolutionary process of IFS-CoCo. Algorithm 2 shows a basic pseudocode of the model proposed. In the following we describe the instructions enumerated from 1 to 9:

Algorithm 2. IFS-CoCo algorithm

```
1  Generate ISPopulation,FSPopulation and IFSPopulation
   Randomly;
2  Select initial bestISArray, bestFSArray and bestIFSArray;
3  Evaluate all populations in the multiclassifier;
4  Select bestISArray, bestFSArray and bestIFSArray from each
   population;
5  while evaluations < max_evaluations do
6    Select best classifier in last generation;
7    Do a CHC Generation on every population;
8    Evaluate the individuals of every population;
9    Update bestISArray, bestFSArray and bestIFSArray if a
   better global solution has been found;
10 end
   Output: bestISArray, bestFSArray and bestIFSArray
```

- Instruction 1 generates the three initial populations. This step includes the random generation of the chromosomes (all of its genes are valued at either 0 or 1, with equal probability), and an initial evaluation of the quality of each chromosome. This basic evaluation consists of building the basic classifier defined by the chromosome (IS classifier, FS classifier or IFS classifier), and the extraction of its related accuracy. Because no use of the general fitness function is made, these *fake* evaluations are not counted into the limit.
- Instruction 2 selects the best individual of each population. With them, every chromosome can be evaluated with the general fitness function, in order to assign them a real fitness value.
- In instruction 3, this evaluation is done by grouping every chromosome with the two chromosomes selected of the other populations (e.g. chromosomes of IS population will employ FS population and IFS population best individuals as partners), and using then the fitness function.
- When the evaluation process is finished, instruction 4 selects the best performing individual of each population.
- Instruction 5 conducts the coevolutionary process.
  - In instruction 6, the best performing individual of each population is selected to help in the task of building the multiclassifiers.
  - Instruction 7 performs a single generation over each population, in an arbitrary order (e.g., IS population, FS population and IFS population), by employing the general EA (CHC, in this case) and the multiclassifier based fitness function.
  - Instruction 8 evaluates the individuals of every population.
  - Instruction 9 concludes a generation, updating the best global solution if a better fitness score has been found. The three chromosomes employed to get this elite solution are saved.

When a fixed number of evaluations run out, the evolutionary process is finished. The algorithm returns the best global solution

found, represented by the best chromosome found in each population.

At the end of the coevolutionary process, the final IFS-CoCo based classifier can be built based on the output chromosomes. This multiclassifier will work in the same manner as all the multiclassifiers employed in the coevolutionary process.

4. Experimental framework

This section shows the details of the experimental framework. Section 4.1 presents the classification problems employed. Section 4.2 summarizes the algorithms employed in the comparison. Section 4.3 describes the parameters employed in each method. Section 4.4 discuss the performance measures employed to evaluate our proposal. Finally, Section 4.5 discusses the statistical tests employed to analyze the results.

4.1. Classification problems

To check the performance of IFS-CoCo, we have used 18 data sets taken from the UCI Machine Learning Database Repository [76]. Table 1 shows their main characteristics. For each data set the number of examples, attributes and classes of the problem described are shown.

Additionally, we have selected a second set of 6 high dimensional data sets (with more than 35 features), of higher size, to perform a second study about the behavior of our approach when the size of the problem increases. All of them have been also taken from the UCI Machine Learning Database Repository [76], except the *Texture data set*, which belongs to the ELENA project.[2] Table 2 shows its characteristics.

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 improve the classification power of the 1-NN rule.

4.2. Algorithms for evaluation

IFS-CoCo will be compared with several evolutionary data reduction algorithms, which manage IS, FS or IFS with the 1-NN rule employed as a baseline classifier.

The concrete algorithms employed are:

- IS algorithms:
  - IS-CHC: CHC algorithm performing IS [13].
  - IS-SSGA: SSGA algorithm performing IS [13].
  - IS-GGA: GGA algorithm performing IS [13].
- FS algorithms:
  - FS-CHC: CHC algorithm performing FS [19].
  - FS-SSGA: SSGA algorithm performing FS.
  - FS-GGA: GGA algorithm performing FS.
- IFS algorithms:
  - IFS-CHC: CHC algorithm performing IS and FS.
  - IGA: Intelligent GA [63].
  - HGA: Hybrid GA [64].
- 1-NN: We compare 1-NN as the basic baseline with all data sets.

CHC, SSGA and GGA based implementations are based on basic evolutionary search processes by the original algorithms, by using the same chromosome representation as the basic population of our model.

[2] ftp://ftp.dice.ucl.ac.be/pub/neural-nets/ELENA/databases

Table 1
UCI Data sets used in our experiments.

Data set	Examples	Attributes	Classes
Aut	205	25	6
Bal	625	4	3
Bupa	345	6	2
Car	1728	6	4
Cleveland	303	13	5
Dermat	366	34	6
German	1000	20	2
Glass	214	9	7
Housevotes	435	16	2
Iris	150	4	3
Mammograph	961	5	2
Pima	768	8	2
Sonar	208	60	2
Spectfheart	267	44	2
Tic-tac-toe	958	9	2
Vehicle	846	18	4
Wisconsin	699	9	2
Zoo	101	16	7

Table 2
High dimensional data sets employed.

Data set	Examples	Attributes	Classes
Chess	3196	36	2
Movement-Libras	360	90	15
Satimage	6435	36	7
Spambase	4597	57	2
Splice	3190	60	3
Texture	5500	40	11

A wider description of all the comparison algorithms can be found in the Appendix A of this contribution.

4.3. Parameters

The most important parameter of IFS-CoCo and the rest of comparison algorithms is the number of evaluations of the fitness function allowed before stopping the search process. We have selected to employ 10,000 evaluations because it is a classical limit employed to test the performance of the majority of EAs for IS [13], which should allow every algorithm to converge in most of the problems used.

The rest parameters used by IFS-CoCo are:

- Population size: 50 (for each population)
- α weighting factor: 0.6
- β weighting factor: 0.99
- *prob0to1* on HUX: 0.25

CHC based algorithms uses the same parameters (including the HUX modified probability), incorporating the fitness function weights when it is necessary.

SSGA and GGA based algorithms also use the same parameters, but they employ standard cross and mutation operators. Their probabilities are:

- Crossing probability (SSGA): 1.0.
- Crossing probability (GGA): 0.6.
- Mutation probability from 0 to 1 (instances): 0.001.

- Mutation probability from 1 to 0 (instances): 0.01.
- Mutation probability (features): 0.01.

IGA parameters are:

- Population size: 50 (for each population).
- Mutation probability 0 to 1 (instances): 0.001.
- Mutation probability 0 to 1 (features): 0.01.
- Mutation probability 1 to 0 (features): 0.01.
- α weighting factor: 0.04.

Finally, HGA has defined 23 parameters. The main parameters' values are:

- Population size: 50 (for each population).
- Crossing probability: 0.5.
- Mutation probability: 0.05.
- The rest of the parameters are set to default values (those listed by the authors in [64]).

4.4. Performance measures

To analyze the results obtained in the study, we have employed three performance measures:

Accuracy: We define the accuracy 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 [1,77,78].

Kappa: Is an alternative to classification rating: a method, known for decades, that compensates for random hits [79]. Its original purpose was to measure the degree of agreement or disagreement between two people observing the same phenomenon.

Cohen's kappa can be adapted to classification tasks and its use recommended because it takes random successes into consideration as a standard, in the same way as the AUC measure [80]. Also, it is used in some well-known software packages, such as WEKA [1], SAS, SPSS, etc. An easy way of computing Cohen's kappa is to make use of the resulting confusion matrix in a classification task. Specifically, the Cohen's kappa measure can be obtained using expression (7):

$$kappa = \frac{n\sum_{i=1}^{C} x_{ii} - \sum_{i=1}^{C} x_{i.}x_{.i}}{n^2 - \sum_{i=1}^{C} x_{i.}x_{.i}} \tag{7}$$

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 $x_{.i}$, $x_{i.}$ are the columns and rows total counts, respectively. Cohen's kappa ranges from -1 (total disagreement) through 0 (random classification) to 1 (perfect agreement). Being a scalar, it is less expressive than ROC curves when applied to binary-classification. However, for multi-class problems, kappa is a very useful, yet simple, meter for measuring the accuracy of the classifier while compensating for random successes.

The main difference between classification rating and Cohen's kappa is the scoring of the correct classifications. Classification rate scores all the successes over all classes, whereas Cohen's kappa scores the successes independently for each class and aggregates them. The second way of scoring is less sensitive to randomness caused by a different number of examples in each class, which causes a bias in the learner towards obtaining data-dependent models.

Reduction: The reduction rate is defined as the ratio of data selected by the algorithm. For example, if a given solution only selects half of the instances (or features) of the training set, its reduction rate will be 0.5. If a given solution only selects half of the instances and half of the features, its reduction rate will be 0.75.

It has a strong influence on the efficiency of the solutions obtained, due to the cost of the final classification process performed by the k-NN classifier ($O(N^2 \cdot M)$).

Time: The simplest way to measure the practical efficiency of a method. We will analyze the average time elapsed (in seconds) by every data reduction method in each complete execution (no times are given for 1-NN, since it does not perform a data reduction phase).

4.5. Test for analysis

To complete the experimental study carried out, we have performed a statistical comparison of accuracy between IFS-CoCo and all the evaluation algorithms. In [81,82] a set of simple, safe and robust non-parametric tests for statistical comparisons of classifiers are recommended. One of them is the Wilcoxon signed-ranks test [27,83], which is the test that we have selected to do the comparison.

This is analogous to the paired t-test in non-parametric statistical procedures; therefore it is a pairwise test that aims to detect significant differences between two sample means, that is, the behavior of two algorithms. It is defined as follows: Let d_i be the difference between the performance scores of the two classifiers on i-th out of N_{ds} data sets. The differences are ranked according to their absolute values; average ranks are assigned in the case of ties. Let R^+ be the sum of ranks for the data sets in which the first algorithm outperformed the second, and R^- the sum of ranks for the opposite. Ranks of $d_i = 0$ are split evenly among the sums; if there is an odd number of them, one is ignored:

$$R^+ = \sum_{d_i > 0} rank(d_i) + \frac{1}{2}\sum_{d_i = 0} rank(d_i)$$

$$R^- = \sum_{d_i < 0} rank(d_i) + \frac{1}{2}\sum_{d_i = 0} rank(d_i) \tag{8}$$

Let T be the smaller of the sums, $T = \min(R^+, R^-)$. If T is less than or equal to the value of the distribution of Wilcoxon for N_{ds} degrees of freedom ([84], Table B.12), the null hypothesis of equality of means is rejected; this will mean that a given classifier outperforms their opposite, with the p-value associated.

The Wilcoxon signed ranks test is more sensible than the t-test. It assumes commensurability of differences, but only qualitatively: greater differences still count for more, which is probably desired, but the absolute magnitudes are ignored. From the statistical point of view, the test is safer since it does not assume normal distributions. Also, the outliers (exceptionally good/bad performances of a few data sets) have less effect on the Wilcoxon than on the t-test. The Wilcoxon test assumes continuous differences d_i, therefore they should not be rounded to one or two decimals, since this would decrease the power of the test in the case of a high number of ties.

When the assumptions of the paired t-test are met, Wilcoxon signed-ranks test is less powerful than the paired t-test. On the other hand, when the assumptions are violated, the Wilcoxon test can be even more powerful than the t-test. This allows us to apply it over the means obtained by the algorithms in each data set, without any assumptions about the sample of results obtained.

A complete description of the Wilcoxon signed ranks test and other non-parametric tests for pairwise and multiple comparisons, together with software for their use, can be found in the website available at http://sci2s.ugr.es/sicidm/.

5. Results and analysis

This section presents the results obtained in the experiment study and analyzes them. In addition, we discuss some advances ideas concerning the behavior of our proposal. Section 5.1 shows the results obtained and analyzes them. Section 5.2 analyzes a comparative study between IFS-CoCo and some classical proposals of IS and FS. Section 5.3 presents a study of how to tune the most important parameters of IFS-CoCo. Section 5.4 shows an analysis of the subsets of instances and features selected by the three populations of IFS-CoCo. Section 5.5 performs an analysis of the convergence of the search process. Section 5.6 shows a second study about the behavior of IFS-CoCo when dealing with high dimensional data sets. Finally, Section 5.7 discusses some interesting trends for future work.

5.1. Results obtained

The results obtained by IFS-CoCo are compared in three categories: IS algorithms, FS algorithms, and IFS algorithms. For each category, two tables are shown:

- Tables 3–5 show the average results in accuracy, kappa, reduction and time elapsed, employing a 3×10- fold cross validation scheme (30 trials per data set) with the 18 data sets of the study. The reduction rate shown for IFS-CoCo corresponds to the relevant population in each category, i.e., the reduction rate in IS population for comparison with IS algorithms and so on.
- Tables 6–8 show the results of performing a two-tailed Wilcoxon Signed-Ranks Test [81] with IFS-CoCo against the respective comparison algorithms. For each test, R^+ and R^- values are shown. Final P-values are computed from these values, as we explained in Section 4.5.

Table 3
IFS-CoCo vs IS algorithms.

	IFS-CoCo	IS-CHC	IS-SSGA	IS-GGA	1-NN
Accuracy	**0.8164**	0.7994	0.7822	0.7789	0.7851
Kappa	**0.6361**	0.5759	0.5950	0.6033	0.5768
Reduction	**0.9818**	0.9617	0.9303	0.9370	–
Time	171.31	**17.67**	39.03	49.70	–

Table 4
IFS-CoCo vs FS algorithms.

	IFS-CoCo	FS-CHC	FS-SSGA	FS-GGA	1-NN
Accuracy	**0.8164**	0.7921	0.7449	0.7438	0.7851
Kappa	**0.6361**	0.6064	0.5048	0.4957	0.5768
Reduction	**0.5200**	0.4419	0.4624	0.4671	–
Time	**171.31**	193.14	180.51	180.50	–

Table 5
IFS-CoCo vs IFS algorithms.

	IFS-CoCo	IFS-CHC	IGA	HGA	1-NN
Accuracy	**0.8164**	0.7978	0.6782	0.7993	0.7851
Kappa	**0.6361**	0.5922	0.3439	0.5836	0.5768
Reduction	0.9911	0.9890	**0.9913**	0.4954	–
Time	171.31	**18.31**	121.76	95.58	–

Table 6
Wilcoxon Signed-Ranks Test for IS algorithms.

IS Algorithms	Accuracy			Kappa		
	R^+	R^-	P-value	R^+	R^-	P-value
IFS-CoCo vs IS-CHC	155	16	0.001	138	33	0.021
IFS-CoCo vs IS-SSGA	162	9	0.000	143	28	0.010
IFS-CoCo vs IS-GGA	154	17	0.001	127	44	0.074
IFS-CoCo vs 1-NN	157	14	0.001	143	28	0.010

Table 7
Wilcoxon Signed-Ranks Test for FS algorithms.

FS Algorithms	Accuracy			Kappa		
	R^+	R^-	P-value	R^+	R^-	P-value
IFS-CoCo vs FS-CHC	140.5	30.5	0.014	125	46	0.090
IFS-CoCo vs FS-SSGA	148	23	0.004	145	26	0.008
IFS-CoCo vs FS-GGA	150	21	0.003	152	19	0.002
IFS-CoCo vs 1-NN	157	14	0.001	143	28	0.010

Table 8
Wilcoxon Signed-Ranks Test for IFS algorithms.

IFS Algorithms	Accuracy			Kappa		
	R^+	R^-	P-value	R^+	R^-	P-value
IFS-CoCo vs IFS-CHC	126	45	0.081	142	29	0.012
IFS-CoCo vs IGA	171	0	0.000	171	0	0.000
IFS-CoCo vs HGA	143	28	0.010	149	22	0.004
IFS-CoCo vs 1-NN	157	14	0.001	143	28	0.010

The full results of this experimental study can be viewed in the Appendix B. Table 14 shows the accuracy results in the training and test phases of IFS-CoCo, the IS algorithms and the 1-NN method. Table 15 shows their kappa results. Table 16 shows the reduction rates achieved by every method. And finally, Table 17 shows the average time elapsed in each data set. In a similar way, Tables 18–21 show the results achieved by FS methods, and Tables 22–25 show the results achieved by IFS methods. Tables regarding accuracy and kappa measures also show the standard deviations, and highlight in bold the best results obtained in the test phase.

Reading the results shown in the tables, we can make the following analysis:

- IFS-CoCo achieves the best average result in accuracy in the three categories.
- IFS-CoCo also achieves the best average result in kappa in the three categories. This means that the success in classification accuracy achieved by our proposal is not caused just by randomness, because it is able to outperform the rest of the algorithms in both performance measures.
- IFS-CoCo is able to obtain higher reduction rates than all the remaining algorithms when each of its populations is compared separately.
- The time taken by IFS-CoCo is comparable to the time spent by FS methods. Although isolated IS and IFS methods are quicker, the increase in time complexity of IFS-CoCo (which is caused by the inclusion of the FS population) can be seen as a minor drawback when we take into account the results obtained in the rest of the performance measures.

- In accuracy, IFS-CoCo outperforms statistically all the comparison algorithms with a level of significance $\alpha = 0.01$, excepting FS-CHC (which IFS-CoCo outperforms with a level of significance $\alpha = 0.05$) and IFS-CHC (IFS-CoCo outperforms it with a level of significance $\alpha = 0.1$).
- In kappa, IFS-CoCo outperforms statistically all the comparison algorithms with a level of significance $\alpha = 0.01$, excepting IS-CHC ($\alpha = 0.05$), IS-GGA ($\alpha = 0.1$), FS-CHC ($\alpha = 0.1$) and IFS-CHC ($\alpha = 0.05$).

The employment of coevolution as a way of breaking the search objective down into three isolated tasks (IS, FS and IFS) has been shown to be quite beneficial, allowing IFS-CoCo to perform a more accurate selection of the relevant data to improve the classification results.

The cooperation between individuals of different populations has allowed our model to better refine the initial data, discarding more noisy and irrelevant instances and features (e.g., some instances which may be relevant if employed with an IS scheme, have become irrelevant with the addition of FS and IFS reduced sets, thus they can be removed safely, improving the generalization capabilities of the classifier). This affirmation is supported by the fact that the reduction rates achieved by each of the populations of IFS-CoCo are slightly higher than the rates of the remaining algorithms. This result confirms that our model can perform a more aggressive reduction of the training data without harming (and even increasing) the generalization capabilities of the 1-NN rule.

5.2. Comparison with classical approaches

To further demonstrate the benefits of our approach, we have performed a second comparison, between IFC-CoCo and some classical non-evolutionary methods for IS and FS.

The methods employed are:

- *DROP3*: A decremental IS procedure proposed in [31]. It performs a noise filtering phase and an instance removal phase, where instances are removed if they do not harm the classification accuracy.
- *ICF*: Another decremental IS procedure, proposed in [85]. It also performs a noise filtering phase before starting the instance removal phase. In its second phase, ICF selects some instances to remove, employing two concepts: Reachability and coverage.
- *Relief*: A filter-based FS method, proposed in [86]. Relief selects features that are statistically relevant, based on how the features represent the decision boundaries of data (employing Euclidean distance). The method is not applied to the entire training set. Instead, a sample (of fixed size) of the training set is extracted to perform the FS procedure.
- *LVW*: A common Las Vegas wrapper-based FS method [87]. This simple method generates a fixed number of random solutions, and tests them by employing the *k*-NN classifier in the training data.

Table 9
IFS-CoCo vs Classical algorithms.

	IFS-CoCo	DROP3	ICF	Relief	LVW
Accuracy	**0.8164**	0.7553	0.7317	0.7472	0.7753
Kappa	**0.6361**	0.5147	0.4880	0.4862	0.5652
Reduction (IS)	**0.9818**	0.8281	0.7328	–	–
Reduction (FS)	**0.5200**	–	-	0.4004	0.3704
Time	171.31	0.44	**0.11**	0.43	150.55

Table 10
Wilcoxon Signed-Ranks Test for Classical algorithms.

IS Algorithms	Accuracy			Kappa		
	R^+	R^-	P-value	R^+	R^-	P-value
IFS-CoCo vs DROP3	169	2	0.001	171	0	0.000
IFS-CoCo vs ICF	171	0	0.000	169	2	0.001
IFS-CoCo vs Relief	150	21	0.003	146	25	0.007
IFS-CoCo vs LVW	148	23	0.005	132	39	0.043

The relevant parameters employed for these methods are:

- Relief: Size Sample: 100. Relevance Threshold: 0.2.
- LVW: Number of solutions: 10000.

Table 9 shows the average results obtained in this second study (the full results can be found in Appendix B, in Tables 26–29). Again, we have performed a Wilcoxon signed ranks test to contrast these results (Table 10).

If we analyze the tables, we can observe that IFS-CoCo outperforms the rest of the proposals in terms of accuracy, kappa and reduction rates. Our approach is able to select better reduced training sets for the 1-NN classifier than the classical ones, enabling it to perform quicker and more accurate classification process, thanks to the higher reduction rates obtained.

However, it is still possible to argue that our approach is slower than the classical ones (except for LVW, which also employs the 1-NN classifier to compute its fitness function). Although this may be seen as a drawback, we can point out that it is not too important if we take into consideration the high reduction rates achieved by our approach. Thanks to its high reduction capabilities, IFS-CoCo will be able to perform a faster classification process of the test set (which, in real life, is usually the most critical phase in terms of time consumption).

The results of the Wilcoxon Signed Ranks test confirm that IFS-CoCo greatly outperforms ($\alpha = 0.01$) the rest of the classical proposals, both in accuracy and kappa measures (except LVW in kappa measure ($\alpha = 0.05$)).

5.3. Selection of suitable parameters for IFS-CoCo

Although we have defined completely how IFS-CoCo works, an interesting question remains: How can a user select suitable values for the parameters of the algorithm?

Some parameters are similar to those usually employed in most of the existing evolutionary approaches for IS and FS. In this way, the number of evaluations of the fitness function (10,000), and the size of the populations (50), can be selected, assuming that those values will work well in most of the problems presented.

However, there are other parameters of IFS-CoCo which cannot be selected by this way. The first of them, the *prob0to1* is easier to set. Experimentally, it is possible to find that this parameter does not have a great impact on the results if it is kept at a reasonable 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, thus producing solutions with high reduction rates but very low results in accuracy due to the impossibility of CHC selecting enough data to represent the initial training set in a suitable way. On the other hand, a value higher than 0.5 will diminish the effect of the modified HUX-cross operator, thus producing solutions with lower reduction rates. Consequently, we have defined $prob0to1 = 0.25$ as a optimal set up.

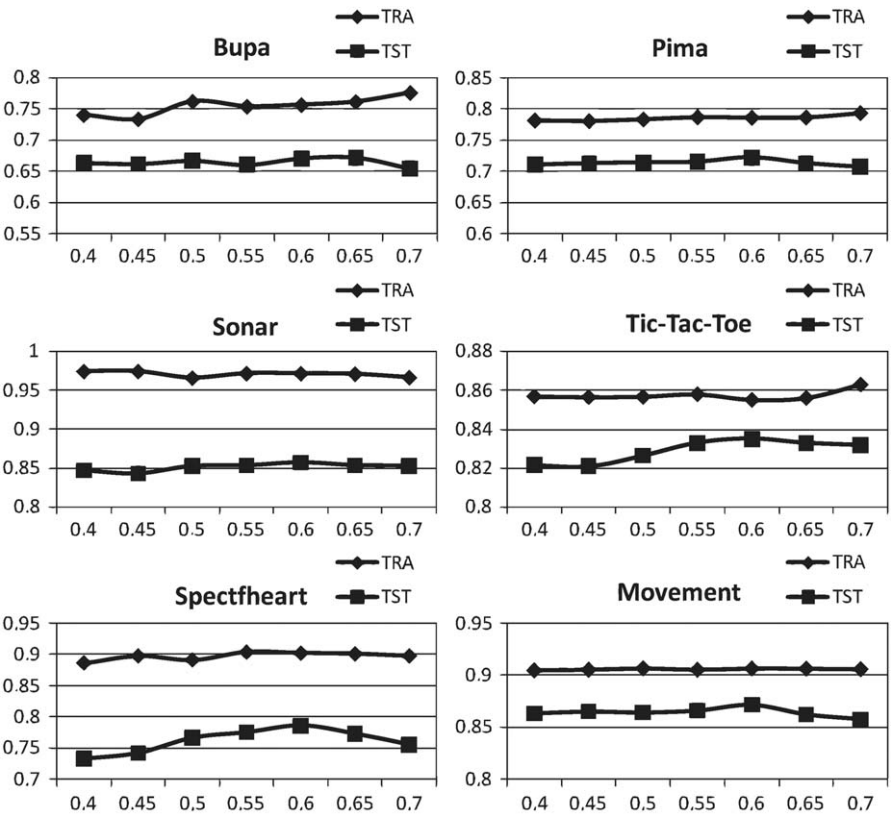

Fig. 3. Results in accuracy by employing different values (0.4–0.7) of the α parameter.

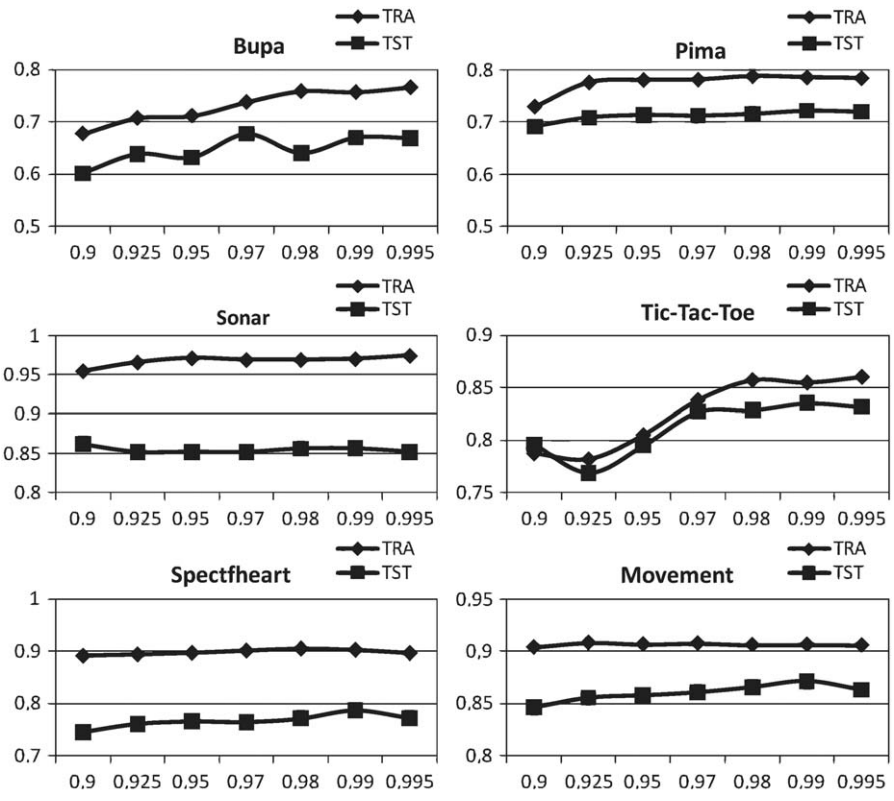

Fig. 4. Results in accuracy by employing different values (0.9–0.995) of the β parameter.

The most influential parameters of IFS-CoCo are α and β. Both define the behavior of the fitness function and the importance of the accuracy and reduction objectives in the search.

The α parameter defines the weight of the instances' reduction rate in the fitness function. The starting point here is $\alpha = 0.5$, because it is the value employed in the majority of evolutionary proposals for IS that incorporates the reduction rate to its fitness functions [13]. To tune it, we have selected six representative data sets (*Bupa*, *Pima*, *Sonar*, *Tic-tac-toe*, *Spectfheart* and *Movement* (Movement-libras)) and tested the effects on test accuracy by changing the value of the parameter to some values between 0.4 and 0.7. The results of the test are shown in Fig. 3 (the *X*-axis represents the possible values of α, and the *X*-axis represents the average test accuracy achieved).

As can be seen in the graphics, an optimal value for α is 0.6. Lower values lead to slightly worse results in classification accuracy, while greater values lead to equal results. However, the reduction rates achieved will be greater the lower the parameter is, thus we have selected $\alpha = 0.6$ as a suitable value for IFS-CoCo.

On the other hand, the β parameter defines the weight of the features' reduction rate in the fitness function. This value should be very near to 1, because the removal of one feature from the training set can produce a high decrease in accuracy due to the large amount of data erased. Thus, only noisy features should be removed, keeping nearly irrelevant ones in the training set if their removal could cause a decrease in accuracy.

Again, we have tested the effects in test accuracy by changing the value of the parameter to some values. This time we have varied the value of the parameter between 0.9 and 0.995. The results of the test are shown in Fig. 4 (the *X*-axis represents the possible values of β, and the *Y*-axis represents the average test accuracy achieved).

As can be seen in the graphics, an optimal value for β is 0.99, both in low-dimensional (*Bupa*, *Pima* and *Tic-tac-toe*) and high dimensional (*Sonar*, *Spectfheart* and *Movement*) data sets. Lower values often lead to worse results in classification accuracy, sometimes producing unstable behavior, while greater values lead to equal results. There is no point in increasing β more, because the classification accuracy does not increase, and it could be counterproductive to the objective of achieving a reasonable reduction rate in the features' component.

5.4. Analysis of the subsets selected by IFS-CoCo

Another interesting question is related to the subsets of instances and features selected as the final solution by each of the populations of IFS-CoCo. What is the criterion employed by our approach to select some subsets of features/instances and discard the rest? Is it a stable decision in subsequent trials in the same data set?. This section is devoted to answering these questions, showing the reasons why a feature/instance will be selected or not for a given problem.

IS population: In the field of IS and prototype selection there are several different approaches to distinguishing which instances must be selected in order to obtain the best possible training set for a given problem. For example, classical condensation algorithms (like CNN [32]) often kept the boundary instances while discard the inner ones. By contrast, classical edition algorithms (like ENN [33]) usually smooth the decision frontiers, removing instances which are near to them. Other algorithms employ more sophisticated methods (like ICF [85], which separates the data into smaller clusters).

Evolutionary approaches try to select the most representative instances achieving the highest reduction as possible. The number and type of instances selected may generally depend on the difficulty of the problem tackled and how appropriate is the *k*-NN classifier for it. Evolutionary methods, like CHC [52], are able to find optimized and adaptive solutions selecting the most suitable instances of the training set without being restricted by prior knowledge about the distribution of the data. For example, the graphical representations of data in [13] show us that the subsets of instances selected are very reduced and have an high quality, showing that CHC selects border or internal instances as needed. IFS-CoCo also takes advantage of this, selecting very reduced subsets of instances of high quality. Evolutionary selection looks for a good distribution of decision frontiers using the Voronoi diagrams resulted from *k*-NN. As we know, similar Voronoi diagrams can be obtained with different subsets of instances, thus it is the main reason that justifies the minimum overlap of instances selected between different runs of the algorithm over all the partitions of the data set.

FS population: By contrast to the instances selected by IS population, the features selected by the FS population show stable behavior in most of the problems.

To analyze it, we have compiled the subsets of features selected by the FS population in every data set (excluding those with more than 20 features, for clarity), over the thirty trials carried out in the $3 \times 10-$cross validation scheme employed in the experimental study. These are presented in Table 11. For each data set the total number of features which compose it is given, the average number of features selected per trial, and the number of times (out 30) where every instance has been selected.

Some interesting conclusions can be drawn from this analysis:

- Some features can be marked relevant (being selected many times). It is possible to extract some patterns in most of the

Table 11
Analysis of the features selected by IFS-CoCo (3×10- cross validation scheme).

Data set	Features	#Selected	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Bal	4	1.4	**11**	10	11	10	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Bupa	6	3.7	**19**	3	**25**	**28**	**27**	18	–	-	–	-	–	-	–	-	–	-	–	–
Car	6	5.0	**30**	**30**	0	**30**	**30**	**30**	–	-	–	-	–	-	–	-	–	-	–	–
Cleveland	13	5.9	**17**	9	**26**	**19**	12	8	4	9	**14**	4	**17**	**30**	13	–	-	–	-	–
Glass	9	5.3	**30**	**20**	**27**	11	8	**25**	**30**	11	0	–	-	–	-	–	-	–	-	–
Housevotes	16	5.4	0	11	**25**	**29**	11	7	**23**	0	**17**	3	2	6	13	14	4	3	–	–
Iris	4	1.7	3	0	**29**	**26**	–	-	–	-	–	-	–	-	–	-	–	-	–	–
Mammographic	5	1.1	10	0	5	**17**	0	–	-	–	-	–	-	–	-	–	-	–	-	–
Pima	8	2.8	15	**21**	5	1	8	**21**	9	**21**	–	-	–	-	–	-	–	-	–	–
Tic-tac-toe	9	6.8	**30**	**16**	**29**	14	**30**	13	**30**	**17**	**30**	–	-	–	-	–	-	–	-	–
Vehicle	18	10.3	**29**	14	**28**	0	**30**	**21**	5	**28**	**22**	**19**	**19**	7	13	6	0	0	**27**	**27**
Wisconsin	9	4.9	**25**	**18**	9	5	**17**	**21**	8	**19**	5	–	-	–	-	–	-	–	-	–
Zoo	16	7.0	**11**	**20**	6	**29**	**11**	**28**	9	3	**26**	4	1	9	**30**	3	3	1	–	–

data sets, which allow us to identify what are the most relevant features in a given problem. Thus, it is possible to assume that the more relevant a given feature is, the greater the number of times it will be selected and the more stable this selection will be.

- It is possible to employ some a priori information to explain the patterns found. For example, for the data set Iris it is known that the two most relevant features are the petal length and the petal width (features #3 and #4). Moreover, when testing the acceptability of a car (car data set), it is often more interesting to know its price (feature #1), cost of maintenances (feature #2), number of persons to carry (feature #4), capacity of the luggage boot (feature #5) or safety (feature #6) than to know the number of doors it has (feature #3), which may also be derived in part from the number of persons to carry.- Furthermore, any experienced player in the game of tic-tac-toe will know that the most important positions to win the game are the center (feature #5) and the corners (features #1, #3, #7 and #9), the rest less interesting if you want to win the game or, at least, prevent your opponent from achieving the victory.
- The relevance of the patterns could also be useful to characterize the data sets in terms of data complexity [88,89]. For example, less relevant patterns (like those extracted in balance, cleveland or pima) often lead to a difficult challenge to FS approaches. By contrast, the behavior of most of the IS approaches in these problems is significantly better.

All of these facts can be employed to explain how the behavior of IFS-CoCo will be with respect to the subsets of selected features: When applied to a problem with well-defined structures, where some attributes could be found to be relevant, our approach will exhibit a stable behavior, selecting those relevant features in most of the trials. By contrast, when facing less well defined problems or with complex relationships between attributes, IFS-CoCo's behavior will be less stable, thus having to rely on the subsets selected by IS and IFS populations.

IFS population: The third population of IFS-CoCo shows peculiar behavior: It selects a very reduced number of instances and features, less instances than the IS population, and less features than the FS population.

However, the explanation is straightforward: Compared with the IS population, the IFS population is able to select the best features of the instances currently selected. The removal of noisy features makes it possible to describe the entire training set with fewer points, thus explaining why the IFS population does not need to select as many instances as the IS population.

The situation is similar when compared with the FS population: Having selected only relevant instances, the subsets selected by the IFS population does not need to take into account every relevant feature in the training set, but only a very reduced number of them. However, the concrete set of features selected is strongly influenced by the currently selected instances, thus the resulting set of selected features is not stable between different trials.

5.5. Analysis of convergence

One of the most important issues in the development of any EA is the analysis of the convergence of its population. If the EA does not evolve in time, most of the time it would not be able to obtain suitable solutions.

In what follows, we show a graphical representation of the convergence capabilities of IFS-CoCo (Fig. 5).

To perform this analysis, we have selected two data sets: Car and Sonar, because they have the greatest number of instances and features, respectively, of the experimental study. The graphics show a line representing the fitness value of the best individual of each population of IFS-CoCo. The *X*-axis represents the number of evaluations carried out, and the *Y*-axis represents the fitness value currently achieved.

As can be seen in the graphics, the classical limit of 10,000 evaluations is enough for IFS-CoCo to converge to a stable solution in the largest examples of our study. The interweaving of the fitness values shows how each population cooperates to allow the global algorithm to converge on good solutions, accepting worse local solutions if it is necessary to improve the global result. This trade-off between the fitness value of the populations, which often improves the results that isolated populations could have

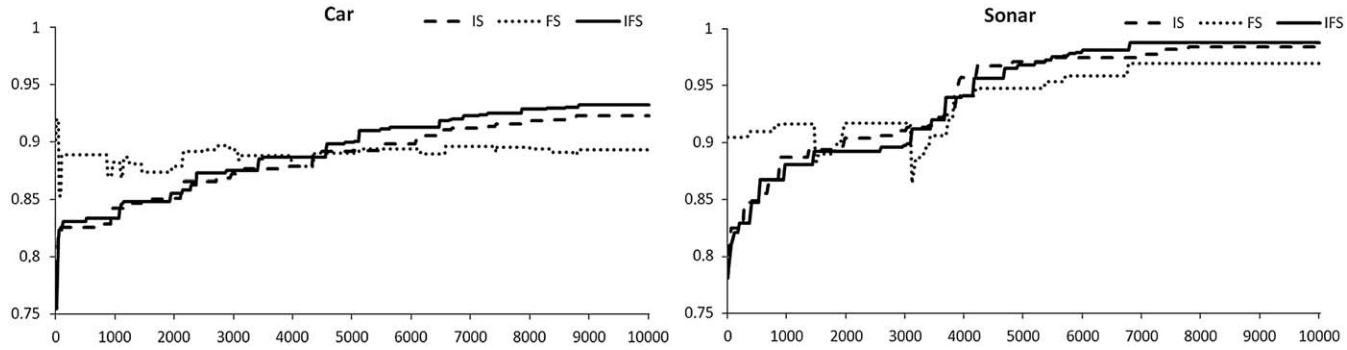

Fig. 5. Map of convergence of IFS-CoCo on Car and Sonar data sets.

Table 12
Average results achieved (high dimensionality data sets).

Alg	IFS-CoCo	IS-CHC	IS-SSGA	IS-GGA	FS-CHC	FS-SSGA	FS-GGA	IFS-CHC	IFS-IGA	IFS-HGA	1-NN
Accuracy	**0.9223**	0.8193	0.8240	0.8371	0.8976	0.8755	0.8741	0.8605	0.8442	0.8632	0.8678
Kappa	**0.8769**	0.7113	0.7391	0.7623	0.8538	0.8104	0.8064	0.7950	0.7669	0.8397	0.7852
Reduction (IS)	0.8372	**0.9628**	0.9454	0.9440	–	-	–	-	–	-	–
Reduction (FS)	**0.5671**	–	–	–	0.5240	0.5369	0.5362	–	-	–	–
Reduction (IFS)	**0.9899**	–	–	–	-	–	-	97.51	98.01	0.6267	–
Time	40914	1461	4294	4615	57747	54819	55523	**1442**	22530	20082	-

Table 13
Wilcoxon Signed-Ranks Test for high dimensional data sets comparison.

	Accuracy			Kappa		
IS Algorithms	R^+	R^-	P-value	R^+	R^-	P-value
IFS-CoCo vs IS-CHC	21	0	0.031	21	0	0.031
IFS-CoCo vs IS-SSGA	21	0	0.031	21	0	0.031
IFS-CoCo vs IS-GGA	21	0	0.031	21	0	0.031
IFS-CoCo vs FS-CHC	19	2	0.093	20	1	0.062
IFS-CoCo vs FS-SSGA	21	0	0.031	21	0	0.031
IFS-CoCo vs FS-GGA	21	0	0.031	21	0	0.031
IFS-CoCo vs IFS-CHC	21	0	0.031	21	0	0.031
IFS-CoCo vs IFS-IGA	21	0	0.031	21	0	0.031
IFS-CoCo vs IFS-HGA	21	0	0.031	21	0	0.031
IFS-CoCo vs 1-NN	20	1	0.062	21	0	0.031

achieved alone, is the main reason for the success of the coevolutionary approach we have employed.

5.6. Analysis of the behavior of IFS-CoCo with high dimensional data sets

Another aspect of our approach that remains unanswered is to test its behavior when dealing with large data sets, specially those with a greater number of attributes. It is important to ensure that our approach is able to tackle these problems having less (or at least, the same) drawbacks than the rest of isolated evolutionary approaches.

To test this behavior, we have employed almost the same experimental framework (see Section 4) that in the standard study, but using the 6 high dimensional data sets described in Table 2.

Table 12 summarizes the results achieved by our approach and all the evolutionary isolated techniques (we also include results from the 1-NN classifier). The best average result for each performance measure is remarked in bold (the full results can be found in Appendix B, in Tables 30–37). In addition, we have performed a Wilcoxon Signed Ranks test to contrast these results (Table 13).

From these tables, we can point out the following facts:

- IFS-CoCo also outperforms the rest of the methods when applied in high dimensionality domains. In fact, it achieves the best precision results, both in accuracy and kappa measures in every data set, except for the Texture data set, where differences are very low. The Wilcoxon test (Table 13) confirm that IFS-CoCo outperforms the rest of the evolutionary proposals, both in accuracy and kappa measures, with a level of significance $\alpha = 0.1$).
- IFS-CoCo also obtains better reduction rates in FS and IFS populations. In the IS population, its reduction rate has decreased.
- The time consumption in higher domains remains the same than in the first study: IFS-CoCo is slower than IS and IFS approaches, but it is faster than FS approaches.

Beyond these results, some general considerations about the behavior of our approach when dealing with greater domains can be obtained. Due to the employment of the CHC algorithm to conduct the search process, the IS population has started to experience some convergence problems when the size of its chromosome has increased from a thousand to more than three thousands genes. This is the reason of the decrease observed in the reduction rates achieved by the IS population, which would had need more evaluations to achieve reduction rates comparable to those achieved by the rest of IS methods (however, note that the IS population only has a third of the total evaluations to accomplish this task, i.e, roughly 3333 evaluations. With a higher number of evaluations, the reduction rates achieved would become the same).

Finally, another consideration must be taken: What happens when the ratio between instances and features (N/M) becomes extreme (very high, or very low)? The answer is that the exact N/M ratio does not have a significant influence in the search process, due to the use of a fixed number of evaluations in each population (a third of them). I.e. the search will not be dominated by the population with a great search space or so, because each population will be able to generate the same number of solutions. However, as we have just discussed, our approach is able to manage an high quality search process, as far as the number of evaluations given were enough. Therefore, if one of these values (N or M) becomes very large and the number of evaluations given does not increases, our approach (as well as the rest of evolutionary ones) will not be able to achieve a high reduction rate in the corresponding population.

5.7. Future trends of work

As a conclusion to the experimental study, we can also point out some interesting topics, which may be taken as starting points for future studies:

Scalability: A promising topic of future work would be to perform a study on the scalability of IFS-CoCo and its application to medium and large size data sets. Although our proposal is not very inefficient in terms of the increment of features and instances ($O(N^2 \cdot M)$), its time complexity could be very high when employed in large scale data sets.

In the field of IS, some strategies have appeared recently to deal with this problems. [90] and [91] are representative examples. However, they are not suitable for IFS-CoCo due to the fact that they only try to split the set of instances, whereas IFS-CoCo requires an explicit treatment of instances, features, and both. Therefore, the development of a new strategy which fits these requirements, and its integration with IFS-CoCo, would be an interesting improvement for our approach.

Weighting schemes: The employment of weighting schemes in an interesting way to preprocess data. Although it falls out of our scope (is not a task of data reduction), it can be employed to assess a given training set, increasing the accuracy rates obtained by the *k*-NN classifier when employing it as training data. It is possible to employ weighting schemes both over the features and the instances of the training set [92].

New coevolutionary-based classifiers could be developed within this scope: Some populations may be focused on obtaining suitable weights for instances and/or features, employing a real-coded EA, whereas other populations may still perform data reduction tasks with binary-coded EAs, as IFS-CoCo does. Indeed, it would be interesting to test the effects of the combination of such different approaches on a concrete method, in terms of the accuracy, reduction and stability of the solutions obtained.

Data complexity: A final topic for future studies falls in the data complexity field [88,89]. This recent field of research tries to characterize the different structures that data sets may show. One of its most interesting applications would be to decide which type of classifier could work better on a given problem.

IFS-CoCo can take advantage of such knowledge: For example, it could be possible to diminish the importance of certain populations, or even disable them, if a given problem requires it (i.e. if a given problem is characterized as very sensitive to the set

of features employed to represent it, our approach would increase the effort applied in its FS population, decreasing the resources spent on the rest of its populations). New improvements of IFS-CoCo may allow the user to specify which behavior IFS-CoCo should show regarding the concrete kind of data set employed.

6. Concluding remarks

In this contribution, we have proposed a new approach based on coevolution, to tackle IS, FS, and IFS problems simultaneously. The employment of a cooperative scheme allows our approach to apply three different data reduction techniques simultaneously, acquiring all its advantages without causing interferences between them, thanks to the employment of coevolution.

The results achieved by IFS-CoCo in the experimental study performed have shown that it offers the best accuracy rates. These results have been contrasted statistically, confirming the hypothesis that it can outperform all the evolutionary methods selected. Moreover, IFS-CoCo has obtained a greater reduction rate than most of the remaining methods, showing its utility as an accurate and effective data reduction technique.

Acknowledgments

We are grateful to the reviewers for identifying essential issues and providing us with important and valuable feedback. We have found the suggestions and comments in the reviews very useful in improving the quality of this paper.

Appendix A. Description of the algorithms employed on the experimental study

To complete the description of the experimental study, this appendix will review the main characteristics of all the comparison algorithms employed. For wider reference about SSGA and GGA, see [13]. For the IGA model, see [63]. For the HGA model, see [64].

A.1. GGA model for IS/FS

GGA is a well-known GA model. The basic idea in GGA is to maintain a population of chromosomes, which encodes plausible solutions to the particular problem that evolves over successive iterations (generations) through a process of competition and controlled variation. Each chromosome in the population has an associated fitness to determine which chromosomes are to be used to form new ones in the competition process. This is called selection. The new ones are created using genetic operators such as crossover and mutation.

The GGA algorithm consists of three operations:

1. evaluation of individual fitness;
2. formation of a gene pool (intermediate population) through selection mechanism;
3. recombination through crossover and mutation operators.

The selection mechanism produces a new population with copies of chromosomes from the previous population. The number of copies received for each chromosome depends on its fitness; chromosomes with higher fitness usually have a greater chance of contributing copies to the new population. Then, the crossover and mutation operators are applied to the copies.

Crossover takes two individuals called parents and produces two new individuals called the offspring by swapping parts of the parents. In its simplest form, the operator works by exchanging substrings after a randomly selected crossover point. The crossover operator is not usually applied to all pairs of chromosomes in the new population. A random choice is made, where the likelihood of crossover being applied depends on probability defined by a crossover rate.

Mutation serves to prevent premature loss of population diversity by randomly sampling new points in the search space. Mutation rates are kept small, however, otherwise the process degenerates into a random search. In the case of bit strings, mutation is applied by flipping one or more random bits in a string with a probability equal to the mutation rate.

Termination may be triggered by reaching a maximum number of generations or by finding an acceptable solution by some criterion.

A.2. SSGA model for IS/FS

SSGA is another well-known GA model, In SSGAs, usually only one or two offspring are produced in each generation. Parents are selected to produce offspring and then a replacement/deletion strategy defines which member of the population will be replaced by the new offspring. The basic algorithm steps of SGA are the following:

1. Select two parents from the population.
2. Create an offspring using crossover and mutation.
3. Evaluate the offspring with the fitness function.
4. Select an individual in, which may be replaced by the offspring.
5. Decide if this individual will be replaced.

In step 4, the replacement strategy chosen has been to replace the worst individuals of the population. In step 5, the replacement condition chosen has been for the replacement to be made only if the new individual is better.

A.3. IGA model for IFS

IGA is an intelligent GA designed to tackle both IS and FS problems simultaneously, by the introduction of a special orthogonal cross operator. In fact, it is an improved GGA model which introduces two new characteristics:

- A rank selection method, which always replace the worst members of the population with the best offspring.
- An orthogonal crossover operator. This is the main feature of IGA, and it was designed in order to improve the selection of the best genes which will be used to form the chromosomes of children.

The high performance of the crossover operator arises from the fact that it replaces the generate-and-test search for children using a random combination of chromosomes with a systematic reasoning search method using an intelligent combination of selecting better individual genes. Thus, the quality of the search procedure is improved.

A.4. HGA model for IFS

HGA is a hybrid GA which employs some local search procedures to improve its results. It performs simultaneous IS and FS procedures, but its objectives are to *minimize* the number

of features selected and to *maximize* the number of instances selected (also trying to increase the accuracy rate).

The HGA algorithm can be divided into three phases:

Phase 1: A pure GA is applied to this phase. It includes an RTS selection scheme and some mechanisms to manage diversity and elitism (including an archive population and a dynamic analysis of the diversity of the population).

Phase 2: By using a histogram of the frequency with which each feature has been selected as present in each chromosome, a feature selection process is carried out, in order to simplify the problem and help the GA to converge.

Phase 3: The GA is applied again to the population. Also, in this phase, some of the children generated in the actual generation are tuned by using local search procedures, both in the features and the instances' search spaces.

Despite the contradictory objectives in the number of instances and features selected, HGA is able to work well on dual IS and FS problems, being a suitable option to tackle these problems.

Appendix B. Full results of the experimental study

As we have mentioned, this appendix contains the full results of the two largest studies performed, the standard one (Section 5.1) and the study with high dimensional data sets.

For the first study, Table 14 shows the average accuracy results (and its standard deviations) in the training and test phases of IFS-CoCo, the IS algorithms and the 1-NN method. The best results in accuracy in the test phase are highlighted in bold. Table 15 shows the results achieved with kappa. Table 16 shows the reduction

Table 14
IFS-CoCo vs IS algorithms (Accuracy in training and test phases).

Alg	IFS-CoCo		IS-CHC		IS-SSGA		IS-GGA		1-NN	
	Training	Test	Training	Test	Training	Test	Training	Test	Training	Test
Aut	85.23 ± 1.22	**77.75 ± 9.25**	83.10 ± 2.60	74.53 ± 7.49	57.91 ± 4.91	59.83 ± 12.22	57.21 ± 3.12	56.07 ± 8.91	75.66 ± 1.21	77.43 ± 6.35
Bal	87.22 ± 1.88	84.95 ± 5.22	78.63 ± 0.43	77.62 ± 2.12	95.93 ± 1.21	86.40 ± 4.32	93.52 ± 3.78	**86.66 ± 2.77**	78.95 ± 0.87	79.04 ± 6.46
Bup	75.66 ± 3.13	**67.05 ± 10.02**	69.63 ± 1.48	60.64 ± 7.37	78.84 ± 4.25	62.28 ± 8.46	68.25 ± 7.63	61.58 ± 7.38	61.22 ± 1.37	61.08 ± 6.88
Car	91.09 ± 0.98	**90.20 ± 2.17**	89.77 ± 1.43	89.56 ± 2.13	84.32 ± 2.75	89.41 ± 2.45	85.55 ± 3.57	86.90 ± 1.99	86.09 ± 0.28	85.65 ± 1.81
Cle	63.35 ± 2.01	**57.99 ± 9.06**	61.18 ± 0.47	55.56 ± 6.62	62.73 ± 8.56	53.60 ± 4.84	60.66 ± 4.46	55.91 ± 6.11	52.77 ± 0.96	53.14 ± 7.45
Der	98.77 ± 0.38	94.93 ± 3.62	98.29 ± 0.72	**96.01 ± 3.53**	91.01 ± 4.31	93.82 ± 4.01	85.59 ± 1.76	95.36 ± 2.57	95.63 ± 0.59	95.35 ± 3.45
Ger	76.94 ± 1.02	**71.80 ± 3.58**	74.98 ± 0.65	70.43 ± 3.01	83.13 ± 2.71	70.87 ± 3.69	80.75 ± 4.16	70.73 ± 4.04	68.97 ± 0.76	70.50 ± 4.25
Gla	77.02 ± 1.88	69.60 ± 10.36	75.60 ± 2.32	67.81 ± 12.54	69.18 ± 5.89	65.80 ± 11.77	60.50 ± 5.89	65.95 ± 13.19	70.77 ± 1.86	**73.61 ± 11.91**
Hou	97.42 ± 0.58	**94.62 ± 4.63**	96.92 ± 0.69	94.47 ± 4.03	89.04 ± 3.90	93.86 ± 4.88	87.91 ± 3.57	93.54 ± 4.52	92.39 ± 0.82	92.16 ± 5.41
Iri	96.09 ± 0.94	95.33 ± 5.33	92.72 ± 0.30	95.33 ± 3.27	95.52 ± 0.27	94.22 ± 4.27	87.97 ± 3.00	**96.00 ± 4.42**	95.48 ± 0.52	93.33 ± 5.16
Mam	84.50 ± 0.57	**83.25 ± 5.19**	84.29 ± 0.75	83.22 ± 3.48	63.88 ± 4.33	79.99 ± 3.94	85.58 ± 5.14	79.85 ± 4.09	73.77 ± 0.87	74.72 ± 5.67
Pim	78.74 ± 0.98	72.27 ± 4.16	75.95 ± 0.66	72.38 ± 5.44	82.44 ± 5.22	72.20 ± 3.59	83.13 ± 4.99	**72.71 ± 4.54**	70.70 ± 0.86	70.33 ± 3.53
Son	97.15 ± 1.17	**85.70 ± 5.99**	94.67 ± 2.66	83.31 ± 8.74	73.60 ± 5.28	79.77 ± 11.83	67.89 ± 5.74	78.49 ± 6.26	86.32 ± 1.08	85.55 ± 7.51
Spe	90.25 ± 0.93	**78.66 ± 5.31**	88.81 ± 1.96	76.16 ± 10.04	79.10 ± 9.75	74.91 ± 8.12	72.10 ± 5.20	76.75 ± 8.31	69.46 ± 1.66	69.70 ± 6.55
Tic	85.48 ± 1.30	**83.51 ± 5.95**	82.10 ± 0.80	82.11 ± 4.82	86.79 ± 3.55	75.09 ± 3.48	78.87 ± 3.69	72.59 ± 3.28	73.13 ± 0.57	73.07 ± 2.56
Veh	74.97 ± 0.99	**70.85 ± 3.35**	73.13 ± 1.14	68.83 ± 5.54	75.97 ± 4.75	66.71 ± 4.38	67.62 ± 6.37	64.11 ± 3.31	69.40 ± 1.13	70.10 ± 5.60
Wis	97.82 ± 0.31	96.09 ± 2.15	85.39 ± 0.24	95.37 ± 2.41	82.62 ± 13.93	96.14 ± 2.02	89.48 ± 9.88	**96.33 ± 2.10**	95.69 ± 0.34	95.57 ± 2.59
Zoo	98.50 ± 1.76	94.97 ± 5.22	98.06 ± 1.65	**95.58 ± 6.49**	82.25 ± 2.31	93.06 ± 6.50	66.30 ± 3.44	92.48 ± 6.40	92.08 ± 0.75	92.81 ± 6.57
Avg.	86.45 ± 1.22	**81.64 ± 5.59**	83.51 ± 1.16	79.94 ± 5.50	79.68 ± 4.88	78.22 ± 5.82	76.60 ± 4.74	77.89 ± 5.23	78.25 ± 0.92	78.51 ± 5.54

Table 15
IFS-CoCo vs IS algorithms (Kappa in training and test phases).

Alg	IFS-CoCo		IS-CHC		IS-SSGA		IS-GGA		1-NN	
	Training	Test	Training	Test	Training	Test	Training	Test	Training	Test
Aut	83.10 ± 1.72	**69.75 ± 8.95**	57.61 ± 2.90	51.08 ± 7.29	84.59 ± 2.90	55.73 ± 11.62	79.33 ± 2.50	53.23 ± 4.81	69.54 ± 1.31	69.41 ± 6.75
Bal	74.94 ± 1.48	75.78 ± 5.12	82.79 ± 1.13	**82.19 ± 2.22**	18.84 ± 0.13	72.24 ± 4.02	61.69 ± 0.63	75.08 ± 1.21	63.08 ± 0.57	63.51 ± 7.06
Bup	51.19 ± 3.53	**31.15 ± 10.32**	44.93 ± 0.78	12.60 ± 7.47	75.97 ± 1.68	21.32 ± 9.16	68.18 ± 1.28	23.29 ± 4.55	19.98 ± 1.37	19.53 ± 6.48
Car	79.89 ± 0.95	77.87 ± 2.13	74.93 ± 2.03	71.96 ± 2.43	80.46 ± 1.73	**79.68 ± 1.75**	75.68 ± 1.73	79.20 ± 3.05	66.47 ± 0.58	65.38 ± 1.71
Cle	35.83 ± 2.03	**30.43 ± 9.02**	37.62 ± 0.23	29.52 ± 6.92	67.19 ± 0.57	27.03 ± 5.14	64.02 ± 0.37	23.64 ± 8.26	26.07 ± 1.26	27.30 ± 7.35
Der	98.48 ± 0.43	91.82 ± 3.92	97.49 ± 0.92	93.16 ± 3.53	36.25 ± 0.42	93.20 ± 4.41	9.28 ± 0.52	93.50 ± 4.51	94.53 ± 0.39	**94.18 ± 2.75**
Ger	38.30 ± 0.82	23.35 ± 3.38	37.71 ± 1.15	21.29 ± 2.61	69.76 ± 0.85	22.84 ± 3.29	60.83 ± 0.45	25.26 ± 3.01	24.82 ± 0.46	**28.00 ± 4.95**
Gla	69.01 ± 1.28	55.44 ± 10.12	65.13 ± 2.02	52.53 ± 12.74	75.15 ± 2.02	53.82 ± 11.37	85.82 ± 2.02	53.43 ± 6.09	60.43 ± 1.86	**64.15 ± 11.81**
Hou	94.48 ± 0.68	**87.00 ± 4.43**	91.99 ± 0.59	86.99 ± 4.73	51.01 ± 0.99	86.94 ± 4.28	50.51 ± 0.89	85.58 ± 3.90	85.78 ± 0.52	86.51 ± 5.31
Iri	96.89 ± 1.04	**94.37 ± 4.83**	98.11 ± 0.30	92.00 ± 3.07	93.42 ± 0.20	93.00 ± 4.27	94.54 ± 0.10	94.00 ± 0.07	93.22 ± 0.82	90.00 ± 5.16
Mam	68.87 ± 0.66	**65.83 ± 6.29**	63.86 ± 0.95	61.86 ± 3.88	59.51 ± 0.85	58.37 ± 4.34	50.66 ± 0.95	58.00 ± 4.03	45.14 ± 0.67	45.73 ± 5.37
Pim	50.53 ± 0.96	36.18 ± 4.36	56.74 ± 0.56	**40.65 ± 4.94**	72.71 ± 0.96	34.75 ± 3.49	60.32 ± 0.66	35.32 ± 5.52	34.40 ± 0.96	33.26 ± 3.93
Son	94.19 ± 1.07	**73.76 ± 5.31**	74.96 ± 1.96	47.55 ± 9.44	73.77 ± 2.76	59.80 ± 12.33	77.39 ± 2.96	72.94 ± 5.08	72.42 ± 1.18	70.77 ± 8.01
Spe	70.36 ± 0.83	26.09 ± 5.79	43.38 ± 2.66	18.38 ± 9.64	41.68 ± 2.06	26.31 ± 8.22	49.81 ± 2.06	**27.49 ± 9.55**	13.76 ± 1.76	12.75 ± 7.05
Tic	69.33 ± 1.30	**60.02 ± 5.69**	49.49 ± 1.00	39.79 ± 5.52	81.14 ± 0.70	43.69 ± 4.08	69.06 ± 0.60	42.10 ± 3.25	27.46 ± 0.27	27.01 ± 1.86
Veh	66.85 ± 0.91	**60.88 ± 3.78**	57.82 ± 1.14	48.83 ± 5.94	81.25 ± 1.04	54.25 ± 4.38	65.66 ± 0.84	54.59 ± 4.95	59.18 ± 1.23	60.10 ± 5.10
Wis	94.40 ± 0.31	**94.37 ± 2.70**	94.75 ± 0.36	91.88 ± 2.21	45.07 ± 0.24	92.78 ± 1.32	50.22 ± 0.35	93.14 ± 14.03	90.39 ± 0.24	90.18 ± 2.19
Zoo	95.16 ± 1.56	90.82 ± 4.94	87.08 ± 0.95	94.41 ± 6.89	95.09 ± 1.75	95.20 ± 5.80	27.35 ± 1.65	**96.23 ± 2.31**	89.55 ± 0.75	90.43 ± 6.77
Avg.	73.99 ± 1.20	**63.61 ± 5.62**	67.58 ± 1.20	57.59 ± 5.64	66.83 ± 1.21	59.50 ± 5.74	61.13 ± 1.14	60.33 ± 4.90	57.57 ± 0.90	57.68 ± 5.53

Table 16
Reduction rates achieved (IS methods).

Algorithms	IFS-CoCo	IS-CHC	IS-SSGA	IS-GGA
Aut	99.62	96.70	86.27	90.10
Bal	97.51	95.06	95.38	94.61
Bup	97.97	94.10	90.84	93.39
Car	95.18	93.85	93.50	90.95
Cle	98.02	95.75	94.76	96.71
Der	99.88	96.60	95.64	96.19
Ger	97.39	95.78	93.81	93.37
Gla	98.60	95.92	89.60	92.49
Hou	99.11	96.69	97.36	97.44
Iri	95.93	95.83	95.01	95.60
Mam	98.53	96.82	97.42	95.03
Pim	98.38	96.02	94.56	94.42
Son	99.36	97.32	87.54	89.99
Spe	98.13	95.95	95.71	96.60
Tic	98.61	97.33	90.62	91.14
Veh	96.98	96.23	90.41	91.27
Wis	99.38	97.25	99.00	98.55
Zoo	98.72	97.85	87.09	88.78
Avg.	98.18	96.17	93.03	93.70

Table 17
Time elapsed (IS methods).

Algorithms	IFS-CoCo	IS-CHC	IS-SSGA	IS-GGA
Aut	17.76	3.86	7.38	6.72
Bal	69.58	13.28	22.64	28.03
Bup	24.27	4.66	10.19	10.16
Car	884.80	97.49	221.54	287.16
Cle	31.87	3.41	7.60	8.03
Der	93.83	7.72	10.42	13.03
Ger	555.26	39.12	93.98	119.79
Gla	14.89	2.95	5.29	5.42
Hou	74.04	6.79	10.73	14.37
Iri	5.46	1.15	1.70	1.63
Mam	223.67	25.42	51.32	73.55
Pim	197.61	20.44	44.31	53.76
Son	44.05	5.62	9.22	9.03
Spe	57.05	4.65	6.94	8.45
Tic	304.43	33.86	97.94	117.97
Veh	300.99	33.10	75.27	96.33
Wis	180.22	13.26	24.39	39.89
Zoo	3.88	1.31	1.65	1.25
Avg.	171.31	17.67	39.03	49.70

Table 18
IFS-CoCo vs FS algorithms (Accuracy in training and test phases).

Alg	IFS-CoCo		FS-CHC		FS-SSGA		FS-GGA		1-NN	
	Training	Test	Training	Test	Training	Test	Training	Test	Training	Test
Aut	85.23 ± 1.22	77.75 ± 9.25	78.28 ± 3.49	77.91 ± 8.79	85.63 ± 8.70	**83.38 ± 11.39**	85.69 ± 8.93	82.90 ± 19.18	75.66 ± 1.21	77.43 ± 6.35
Bal	87.22 ± 1.88	**84.95 ± 5.22**	74.70 ± 7.14	74.79 ± 8.49	75.65 ± 0.87	71.03 ± 6.46	79.65 ± 0.87	72.66 ± 6.46	78.95 ± 0.87	79.04 ± 6.46
Bup	75.66 ± 3.13	**67.05 ± 10.02**	62.22 ± 1.48	62.51 ± 8.98	64.81 ± 1.53	60.58 ± 10.57	64.80 ± 1.59	60.78 ± 10.57	61.22 ± 1.37	61.08 ± 6.88
Car	91.09 ± 0.98	90.20 ± 2.17	90.68 ± 0.35	**90.68 ± 1.43**	82.11 ± 0.28	81.89 ± 1.73	82.11 ± 0.28	81.89 ± 1.73	86.09 ± 0.28	85.65 ± 1.81
Cle	63.35 ± 2.01	**57.99 ± 9.06**	50.25 ± 1.00	50.30 ± 3.11	58.71 ± 0.85	50.38 ± 4.89	58.68 ± 0.97	49.96 ± 5.59	52.77 ± 0.96	53.14 ± 7.45
Der	98.77 ± 0.38	94.93 ± 3.62	93.74 ± 0.41	94.28 ± 3.83	97.71 ± 4.74	94.28 ± 5.23	98.39 ± 1.08	93.17 ± 3.80	95.63 ± 0.59	**95.35 ± 3.45**
Ger	76.94 ± 1.02	**71.80 ± 3.58**	70.60 ± 1.13	69.70 ± 4.58	72.86 ± 1.17	69.23 ± 3.53	72.84 ± 0.72	68.93 ± 4.98	68.97 ± 0.76	70.50 ± 4.25
Gla	77.02 ± 1.88	69.60 ± 10.36	71.80 ± 1.69	71.68 ± 12.58	77.66 ± 1.76	70.39 ± 13.85	77.68 ± 1.74	70.54 ± 13.11	70.77 ± 1.86	**73.61 ± 11.91**
Hou	97.42 ± 0.58	**94.62 ± 4.63**	94.70 ± 0.33	94.24 ± 3.45	81.14 ± 12.61	82.24 ± 13.38	78.44 ± 11.59	79.79 ± 9.91	92.39 ± 0.82	92.16 ± 5.41
Iri	96.09 ± 0.94	95.33 ± 5.33	96.00 ± 0.82	96.00 ± 4.42	94.96 ± 0.80	**96.67 ± 3.33**	94.96 ± 0.80	96.00 ± 3.27	95.48 ± 0.52	93.33 ± 5.16
Mam	84.50 ± 0.57	**83.25 ± 5.19**	75.65 ± 1.40	75.65 ± 5.76	57.92 ± 9.79	54.61 ± 9.48	58.46 ± 8.39	55.72 ± 8.57	73.77 ± 0.87	74.72 ± 5.67
Pim	78.74 ± 0.98	**72.27 ± 4.16**	68.86 ± 0.52	68.00 ± 4.96	71.90 ± 0.49	67.31 ± 5.06	71.90 ± 0.49	67.31 ± 5.06	70.70 ± 0.86	70.33 ± 3.53
Son	97.15 ± 1.17	85.70 ± 5.99	87.98 ± 1.95	86.53 ± 9.23	95.66 ± 0.56	84.48 ± 10.38	96.92 ± 1.44	**87.11 ± 7.65**	86.32 ± 1.08	85.55 ± 7.51
Spe	90.25 ± 0.93	**78.66 ± 5.31**	72.66 ± 3.14	73.51 ± 4.85	86.14 ± 1.42	73.45 ± 7.56	86.61 ± 0.99	72.94 ± 5.91	69.46 ± 1.66	69.70 ± 6.55
Tic	85.48 ± 1.30	**83.51 ± 5.95**	82.78 ± 0.53	82.33 ± 2.46	69.78 ± 1.09	69.52 ± 2.03	70.03 ± 0.69	69.91 ± 2.33	73.13 ± 0.57	73.07 ± 2.56
Veh	74.97 ± 0.99	70.85 ± 3.35	70.58 ± 0.77	70.97 ± 5.14	74.84 ± 0.68	**73.01 ± 3.71**	74.88 ± 0.43	72.38 ± 4.68	69.40 ± 1.13	70.10 ± 5.60
Wis	97.82 ± 0.31	**96.09 ± 2.15**	93.73 ± 0.39	95.26 ± 2.28	96.48 ± 0.48	95.23 ± 2.23	96.47 ± 0.61	95.09 ± 2.23	95.69 ± 0.34	95.57 ± 2.59
Zoo	98.50 ± 1.76	**94.97 ± 5.22**	95.14 ± 1.12	91.39 ± 4.79	61.63 ± 13.11	63.07 ± 14.56	59.63 ± 15.47	61.78 ± 12.30	92.08 ± 0.75	92.81 ± 6.57
Avg.	86.45 ± 1.22	**81.64 ± 5.59**	79.46 ± 1.54	79.21 ± 5.51	78.09 ± 3.38	74.49 ± 7.19	78.23 ± 3.17	74.38 ± 7.07	78.25 ± 0.92	78.51 ± 5.54

Table 19
IFS-CoCo vs FS algorithms (Kappa in training and test phases).

Alg	IFS-CoCo		FS-CHC		FS-SSGA		FS-GGA		1-NN	
	Training	Test	Training	Test	Training	Test	Training	Test	Training	Test
Aut	83.10 ± 1.72	69.75 ± 8.95	81.69 ± 3.19	**72.36 ± 8.49**	72.98 ± 9.00	70.70 ± 11.59	66.13 ± 9.23	58.72 ± 19.48	69.54 ± 1.31	69.41 ± 6.75
Bal	74.94 ± 1.48	**75.78 ± 5.12**	63.08 ± 7.44	63.51 ± 8.19	63.08 ± 0.67	63.51 ± 6.26	63.08 ± 0.77	63.51 ± 6.56	63.08 ± 0.57	63.51 ± 7.06
Bup	51.19 ± 3.53	**31.15 ± 10.32**	29.99 ± 1.18	24.17 ± 8.98	27.01 ± 1.63	18.21 ± 10.27	26.94 ± 1.39	18.21 ± 10.57	19.98 ± 1.37	19.53 ± 6.48
Car	79.89 ± 0.95	77.87 ± 2.13	78.80 ± 0.25	**78.71 ± 1.23**	54.57 ± 0.08	54.25 ± 2.03	54.57 ± 0.28	54.25 ± 1.73	66.47 ± 0.58	65.38 ± 1.71
Cle	35.83 ± 2.03	**30.43 ± 9.02**	34.93 ± 1.30	21.64 ± 3.01	35.13 ± 1.05	27.28 ± 5.09	35.60 ± 0.87	23.44 ± 5.69	26.07 ± 1.26	27.30 ± 7.35
Der	98.48 ± 0.43	91.82 ± 3.92	98.37 ± 0.71	93.87 ± 3.93	95.64 ± 4.54	91.81 ± 5.23	97.80 ± 0.98	92.48 ± 4.10	94.53 ± 0.39	**94.18 ± 2.75**
Ger	38.30 ± 0.82	23.35 ± 3.38	36.30 ± 1.13	26.27 ± 4.78	33.66 ± 1.37	19.29 ± 3.53	36.84 ± 0.52	26.72 ± 4.68	24.82 ± 0.46	**28.00 ± 4.95**
Gla	69.01 ± 1.28	55.44 ± 10.12	70.14 ± 1.39	60.24 ± 12.78	69.84 ± 2.06	62.46 ± 13.85	69.74 ± 1.64	63.55 ± 12.91	60.43 ± 1.86	**64.15 ± 11.81**
Hou	94.48 ± 0.68	**87.00 ± 4.43**	92.28 ± 0.03	84.07 ± 3.75	53.56 ± 12.41	46.72 ± 13.28	53.48 ± 11.39	53.64 ± 9.91	85.78 ± 0.52	86.51 ± 5.31
Iri	96.89 ± 1.04	94.37 ± 4.83	94.00 ± 0.62	93.00 ± 4.12	92.44 ± 0.50	**95.00 ± 3.13**	92.44 ± 0.70	94.00 ± 3.47	93.22 ± 0.82	90.00 ± 5.16
Mam	68.87 ± 0.66	**65.83 ± 6.29**	51.41 ± 1.60	48.68 ± 5.56	21.87 ± 9.69	16.26 ± 9.18	18.49 ± 8.39	12.05 ± 8.67	45.14 ± 0.67	45.73 ± 5.37
Pim	50.53 ± 0.96	**36.18 ± 4.36**	34.59 ± 0.72	25.99 ± 4.76	38.25 ± 0.19	28.01 ± 4.86	38.25 ± 0.49	28.01 ± 4.96	34.40 ± 0.96	33.26 ± 3.93
Son	94.19 ± 1.07	**73.76 ± 5.31**	91.60 ± 1.95	72.57 ± 9.33	91.50 ± 0.76	72.85 ± 10.68	93.23 ± 1.54	71.45 ± 7.65	72.42 ± 1.18	70.77 ± 8.01
Spe	70.36 ± 0.83	26.09 ± 5.79	62.40 ± 3.04	22.87 ± 5.15	59.17 ± 1.12	**28.10 ± 7.76**	59.80 ± 0.69	19.28 ± 6.11	13.76 ± 1.76	12.75 ± 7.05
Tic	69.33 ± 1.30	**60.02 ± 5.69**	59.31 ± 0.43	57.48 ± 2.46	15.37 ± 1.09	15.94 ± 1.73	18.30 ± 0.89	17.45 ± 2.33	27.46 ± 0.27	27.01 ± 1.86
Veh	66.85 ± 0.91	60.88 ± 3.78	66.66 ± 0.97	61.68 ± 5.44	66.38 ± 0.88	61.21 ± 3.41	66.59 ± 0.43	**62.15 ± 4.98**	59.18 ± 1.23	60.10 ± 5.10
Wis	94.40 ± 0.31	**94.37 ± 2.70**	92.33 ± 0.39	89.52 ± 2.28	92.43 ± 0.58	89.22 ± 2.23	92.26 ± 0.51	89.22 ± 2.03	90.39 ± 0.24	90.18 ± 2.19
Zoo	95.16 ± 1.56	90.82 ± 4.94	97.70 ± 1.02	**94.94 ± 4.59**	45.71 ± 13.01	47.76 ± 14.86	45.26 ± 15.17	44.18 ± 12.30	89.55 ± 0.75	90.43 ± 6.77
Avg.	73.99 ± 1.20	**63.61 ± 5.62**	68.64 ± 1.52	60.64 ± 5.49	57.15 ± 3.37	50.48 ± 7.17	57.16 ± 3.10	49.57 ± 7.12	57.57 ± 0.90	57.68 ± 5.53

Table 20
Reduction rates achieved (FS methods).

Algorithms	IFS-CoCo	FS-CHC	FS-SSGA	FS-GGA
Aut	69.20	68.27	66.39	67.44
Bal	65.00	3.33	4.55	4.21
Bup	38.33	30.00	31.56	29.34
Car	16.67	16.67	20.04	24.32
Cle	54.62	48.72	48.89	48.32
Der	55.88	56.37	54.85	55.88
Ger	43.00	42.33	42.67	41.99
Gla	41.11	44.07	42.01	45.37
Hou	66.25	62.29	63.45	68.69
Iri	57.50	40.00	52.25	48.75
Mam	78.00	50.00	58.64	61.50
Pim	65.00	53.75	57.32	55.67
Son	57.17	59.50	58.43	59.25
Spe	59.09	55.76	56.71	56.05
Tic	24.44	22.22	27.35	26.34
Veh	42.78	45.52	46.23	44.57
Wis	45.56	41.11	40.59	40.87
Zoo	56.32	55.45	60.34	62.28
Avg.	52.00	44.19	46.24	46.71

Table 21
Time elapsed (FS methods).

Algorithms	IFS-CoCo	FS-CHC	FS-SSGA	FS-GGA
Aut	17.76	25.10	29.53	28.73
Bal	69.58	88.33	76.04	75.02
Bup	24.27	32.71	30.68	30.88
Car	884.80	963.64	658.31	694.10
Cle	31.87	37.46	52.00	50.72
Der	93.83	96.62	108.50	109.44
Ger	555.26	607.66	597.15	577.39
Gla	14.89	20.03	20.00	19.49
Hou	74.04	97.19	89.55	95.14
Iri	5.46	6.91	5.91	5.88
Mam	223.67	236.26	289.70	289.98
Pim	197.61	231.68	207.48	201.59
Son	44.05	58.34	62.08	60.75
Spe	57.05	71.35	79.53	74.89
Tic	304.43	361.16	360.59	366.98
Veh	300.99	357.91	373.93	367.86
Wis	180.22	178.57	202.59	195.08
Zoo	3.88	5.53	5.60	5.17
Avg.	171.31	193.14	180.51	180.50

Table 22
IFS-CoCo vs IFS algorithms (Accuracy in training and test phases).

Alg	IFS-CoCo		IFS-CHC		IGA		HGA		1-NN	
	Training	Test	Training	Test	Training	Test	Training	Test	Training	Test
Aut	85.23 ± 1.22	77.75 ± 9.25	71.89 ± 1.80	70.03 ± 10.05	71.92 ± 6.93	68.41 ± 13.00	78.05 ± 3.99	**78.01 ± 11.72**	75.66 ± 1.21	77.43 ± 6.35
Bal	87.22 ± 1.88	84.95 ± 5.22	90.20 ± 0.67	**88.32 ± 2.34**	52.00 ± 8.15	52.52 ± 9.58	83.85 ± 5.54	82.76 ± 5.82	78.95 ± 0.87	79.04 ± 6.46
Bup	75.66 ± 3.13	67.05 ± 10.02	73.84 ± 1.55	**69.29 ± 8.32**	59.80 ± 8.99	54.67 ± 7.75	66.18 ± 1.98	65.47 ± 8.16	61.22 ± 1.37	61.08 ± 6.88
Car	91.09 ± 0.98	**90.20 ± 2.17**	90.73 ± 0.80	89.35 ± 1.77	88.79 ± 1.35	86.85 ± 0.36	89.54 ± 0.55	88.86 ± 2.95	86.09 ± 0.28	85.65 ± 1.81
Cle	63.35 ± 2.01	57.99 ± 9.06	62.54 ± 0.57	**58.20 ± 4.44**	42.26 ± 11.13	42.72 ± 7.13	58.73 ± 1.20	56.23 ± 4.74	52.77 ± 0.96	53.14 ± 7.45
Der	98.77 ± 0.38	94.93 ± 3.62	97.33 ± 0.40	**95.52 ± 3.28**	95.32 ± 9.73	94.31 ± 12.12	97.12 ± 0.51	95.42 ± 6.01	95.63 ± 0.59	95.35 ± 3.45
Ger	76.94 ± 1.02	71.80 ± 3.58	75.09 ± 1.62	**72.77 ± 2.21**	73.47 ± 4.31	70.80 ± 6.18	72.43 ± 0.93	70.77 ± 8.69	68.97 ± 0.76	70.50 ± 4.25
Gla	77.02 ± 1.88	69.60 ± 10.36	73.97 ± 2.31	67.30 ± 10.25	52.04 ± 8.23	57.13 ± 10.63	72.34 ± 1.39	70.53 ± 10.77	70.77 ± 1.86	**73.61 ± 11.91**
Hou	97.42 ± 0.58	**94.62 ± 4.63**	94.79 ± 0.76	93.99 ± 3.62	66.79 ± 13.27	67.65 ± 16.07	94.10 ± 0.33	93.65 ± 4.38	92.39 ± 0.82	92.16 ± 5.41
Iri	96.09 ± 0.94	**95.33 ± 5.33**	96.94 ± 0.66	94.89 ± 4.99	89.92 ± 19.68	89.33 ± 10.83	96.75 ± 0.81	95.16 ± 6.57	95.48 ± 0.52	93.33 ± 5.16
Mam	84.50 ± 0.57	**83.25 ± 5.19**	82.11 ± 0.81	81.21 ± 5.54	69.40 ± 11.68	68.99 ± 13.03	81.55 ± 1.30	80.36 ± 5.14	73.77 ± 0.87	74.72 ± 5.67
Pim	78.74 ± 0.98	72.27 ± 4.16	78.76 ± 0.54	73.67 ± 4.51	57.63 ± 8.98	58.64 ± 6.88	77.37 ± 0.66	**74.17 ± 5.59**	70.70 ± 0.86	70.33 ± 3.53
Son	97.15 ± 1.17	**85.70 ± 5.99**	85.40 ± 2.24	75.61 ± 9.42	81.21 ± 12.07	78.78 ± 8.60	84.34 ± 1.75	77.42 ± 10.43	86.32 ± 1.08	85.55 ± 7.51
Spe	90.25 ± 0.93	**78.66 ± 5.31**	84.56 ± 1.60	76.71 ± 6.05	74.44 ± 11.20	71.91 ± 9.90	79.87 ± 4.14	72.05 ± 6.92	69.46 ± 1.66	69.70 ± 6.55
Tic	85.48 ± 1.30	**83.51 ± 5.95**	78.44 ± 2.30	76.38 ± 2.45	55.16 ± 2.38	65.35 ± 1.32	78.18 ± 0.23	77.96 ± 4.78	73.13 ± 0.57	73.07 ± 2.56
Veh	74.97 ± 0.99	70.85 ± 3.35	72.13 ± 0.74	67.53 ± 3.89	53.89 ± 10.77	54.33 ± 10.76	71.21 ± 0.99	**70.98 ± 2.38**	69.40 ± 1.13	70.10 ± 5.60
Wis	97.82 ± 0.31	**96.09 ± 2.15**	97.18 ± 0.32	95.52 ± 1.96	67.89 ± 4.27	68.88 ± 3.21	97.69 ± 3.94	95.69 ± 3.12	95.69 ± 0.34	95.57 ± 2.59
Zoo	98.50 ± 1.76	**94.97 ± 5.22**	94.76 ± 0.98	89.72 ± 7.45	64.11 ± 2.76	69.50 ± 4.53	95.15 ± 1.62	93.17 ± 6.90	92.08 ± 0.75	92.81 ± 6.57
Avg.	86.45 ± 1.22	**81.64 ± 5.59**	83.37 ± 1.15	79.78 ± 5.14	67.56 ± 8.66	67.82 ± 8.44	81.91 ± 1.77	79.93 ± 6.39	78.25 ± 0.92	78.51 ± 5.54

Table 23
IFS-CoCo vs IFS algorithms (Kappa in training and test phases).

Alg	IFS-CoCo		IFS-CHC		IGA		HGA		1-NN	
	Training	Test	Training	Test	Training	Test	Training	Test	Training	Test
Aut	83.10 ± 1.72	**69.75 ± 8.95**	63.81 ± 1.60	60.06 ± 10.25	38.44 ± 6.83	27.46 ± 13.00	83.81 ± 3.79	62.77 ± 11.82	69.54 ± 1.31	69.41 ± 6.75
Bal	74.94 ± 1.48	75.78 ± 5.12	81.87 ± 0.57	**78.65 ± 2.04**	54.05 ± 8.05	32.38 ± 9.48	76.85 ± 5.74	69.34 ± 5.82	63.08 ± 0.57	63.51 ± 7.06
Bup	51.19 ± 3.53	31.15 ± 10.32	44.15 ± 1.45	**34.65 ± 8.32**	27.64 ± 9.19	20.84 ± 7.95	40.38 ± 1.68	27.65 ± 8.36	19.98 ± 1.37	19.53 ± 6.48
Car	79.89 ± 0.95	**77.87 ± 2.13**	79.12 ± 0.80	74.81 ± 1.77	62.14 ± 1.35	22.61 ± 0.56	78.32 ± 0.75	74.92 ± 2.85	66.47 ± 0.58	65.38 ± 1.71
Cle	35.83 ± 2.03	**30.43 ± 9.02**	34.62 ± 0.37	26.02 ± 4.54	19.70 ± 11.33	11.01 ± 7.03	33.12 ± 1.30	22.92 ± 4.64	26.07 ± 1.26	27.30 ± 7.35
Der	98.48 ± 0.43	91.82 ± 3.92	96.61 ± 0.20	**95.20 ± 3.18**	63.85 ± 9.63	53.62 ± 11.82	96.73 ± 0.31	94.70 ± 5.91	94.53 ± 0.39	94.18 ± 2.75
Ger	38.30 ± 0.82	23.35 ± 3.38	34.41 ± 1.72	27.07 ± 2.41	16.70 ± 4.11	14.68 ± 6.28	30.11 ± 0.73	22.34 ± 8.79	24.82 ± 0.46	**28.00 ± 4.95**
Gla	69.01 ± 1.28	55.44 ± 10.12	62.84 ± 2.61	51.18 ± 10.55	45.25 ± 7.93	30.02 ± 10.43	61.84 ± 1.09	53.18 ± 11.07	60.43 ± 1.86	**64.15 ± 11.81**
Hou	94.48 ± 0.68	87.00 ± 4.43	89.70 ± 0.46	**87.21 ± 3.52**	62.31 ± 12.97	55.60 ± 15.77	88.98 ± 0.53	87.02 ± 4.58	85.78 ± 0.52	86.51 ± 5.31
Iri	96.89 ± 1.04	**94.37 ± 4.83**	96.22 ± 0.76	92.00 ± 4.89	80.09 ± 19.58	76.00 ± 10.53	97.01 ± 0.51	91.87 ± 6.47	93.22 ± 0.82	90.00 ± 5.16
Mam	68.87 ± 0.66	**65.83 ± 6.29**	64.20 ± 1.11	61.54 ± 5.84	46.72 ± 11.38	47.39 ± 12.73	60.73 ± 1.10	52.83 ± 5.34	45.14 ± 0.67	45.73 ± 5.37
Pim	50.53 ± 0.96	36.18 ± 4.36	51.61 ± 0.44	38.62 ± 4.71	40.21 ± 9.08	25.20 ± 6.98	50.56 ± 0.66	**40.12 ± 5.59**	34.40 ± 0.96	33.26 ± 3.93
Son	94.19 ± 1.07	**73.76 ± 5.31**	69.85 ± 2.44	46.83 ± 9.12	51.58 ± 12.17	19.83 ± 8.70	72.85 ± 1.55	47.40 ± 10.73	72.42 ± 1.18	70.77 ± 8.01
Spe	70.36 ± 0.83	**26.09 ± 5.79**	43.98 ± 1.80	16.08 ± 6.35	33.24 ± 11.50	8.48 ± 9.90	38.76 ± 3.84	11.03 ± 6.92	13.76 ± 1.76	12.75 ± 7.05
Tic	69.33 ± 1.30	**60.02 ± 5.69**	52.44 ± 2.50	43.92 ± 2.65	39.51 ± 2.08	23.92 ± 1.12	53.11 ± 0.23	49.25 ± 4.68	27.46 ± 0.27	27.01 ± 1.86
Veh	66.85 ± 0.91	60.88 ± 3.78	64.01 ± 0.84	57.10 ± 3.99	48.40 ± 11.07	47.20 ± 10.46	61.97 ± 0.79	**61.48 ± 2.58**	59.18 ± 1.23	60.10 ± 5.10
Wis	94.40 ± 0.31	**94.37 ± 2.70**	94.25 ± 0.12	90.32 ± 1.86	81.25 ± 4.57	81.71 ± 3.31	94.01 ± 3.74	90.92 ± 3.02	90.39 ± 0.24	90.18 ± 2.19
Zoo	95.16 ± 1.56	**90.82 ± 4.94**	92.87 ± 1.28	84.77 ± 7.35	64.90 ± 2.66	21.07 ± 4.33	93.17 ± 1.42	90.77 ± 6.60	89.55 ± 0.75	90.43 ± 6.77
Avg.	73.99 ± 1.20	**63.61 ± 5.62**	67.59 ± 1.17	59.22 ± 5.19	48.67 ± 8.64	34.39 ± 8.35	67.35 ± 1.66	58.36 ± 6.43	57.57 ± 0.90	57.68 ± 5.53

Table 24
Reduction rates achieved (IFS methods).

Algorithms	IFS-CoCo	IFS-CHC	IGA	HGA
Aut	99.19	98.87	98.90	72.34
Bal	98.85	98.42	98.78	11.34
Bup	99.37	99.21	99.05	33.45
Car	98.96	97.76	98.65	25.77
Cle	99.50	99.45	99.52	59.20
Der	99.33	99.14	99.23	15.34
Ger	99.85	99.89	99.87	54.05
Gla	98.30	97.47	97.95	51.04
Hou	99.87	99.88	99.85	63.09
Iri	98.17	97.91	97.97	45.67
Mam	99.86	99.86	99.84	60.85
Pim	99.74	99.67	99.61	63.87
Son	99.72	99.68	99.54	70.01
Spe	99.88	99.87	99.86	62.41
Tic	99.33	99.02	99.43	30.45
Veh	99.11	99.03	98.85	53.82
Wis	99.68	99.74	99.70	49.56
Zoo	95.32	95.40	97.67	69.43
Avg.	99.11	98.90	99.13	49.54

Table 25
Time elapsed (IFS methods).

Algorithms	IFS-CoCo	IFS-CHC	IGA	HGA
Aut	17.76	3.90	13.21	14.27
Bal	69.58	17.16	77.67	39.72
Bup	24.27	5.04	22.91	15.64
Car	884.80	89.31	669.95	421.16
Cle	31.87	4.35	22.67	26.38
Der	93.83	8.37	48.78	58.12
Ger	555.26	40.85	333.55	275.34
Gla	14.89	3.11	8.48	12.84
Hou	74.04	7.71	45.99	41.63
Iri	5.46	1.27	2.70	2.36
Mam	223.67	26.64	190.28	162.85
Pim	197.61	23.80	134.60	101.11
Son	44.05	4.58	21.73	25.49
Spe	57.05	4.88	31.53	59.78
Tic	304.43	33.50	221.41	192.29
Veh	300.99	35.67	226.63	162.20
Wis	180.22	18.11	117.73	107.65
Zoo	3.88	1.33	1.89	1.60
Avg.	171.31	18.31	121.76	95.58

Table 26
IFS-CoCo vs Classical algorithms (Accuracy in training and test phases).

Alg	IFS-CoCo		DROP3		ICF		Relief		LVW	
	Training	Test	Training	Test	Training	Test	Training	Test	Training	Test
Aut	85.23 ± 1.22	77.75 ± 9.25	91.08 ± 1.64	62.29 ± 8.71	92.00 ± 2.43	58.81 ± 6.69	77.29 ± 1.64	78.16 ± 8.54	83.80 ± 3.90	**78.17 ± 9.71**
Bal	87.22 ± 1.88	**84.95 ± 5.22**	88.08 ± 3.64	81.77 ± 3.82	97.99 ± 1.86	70.26 ± 5.52	54.47 ± 1.23	54.40 ± 5.10	78.95 ± 0.87	79.04 ± 6.46
Bup	75.66 ± 3.13	**67.05 ± 10.02**	79.45 ± 3.90	60.86 ± 7.15	68.58 ± 4.09	53.29 ± 9.87	55.36 ± 2.33	52.46 ± 10.38	65.09 ± 1.38	61.37 ± 9.65
Car	91.09 ± 0.98	**90.20 ± 2.17**	92.11 ± 2.04	72.21 ± 4.30	96.73 ± 1.15	78.42 ± 3.44	71.95 ± 1.32	71.65 ± 1.32	82.11 ± 0.28	81.89 ± 1.73
Cle	63.35 ± 2.01	**57.99 ± 9.06**	85.12 ± 3.36	49.47 ± 4.24	80.54 ± 6.26	50.19 ± 5.63	45.98 ± 7.74	42.85 ± 9.53	57.65 ± 5.50	44.88 ± 9.85
Der	98.77 ± 0.38	94.93 ± 3.62	88.02 ± 3.56	92.93 ± 4.20	96.51 ± 2.03	90.52 ± 3.61	96.69 ± 0.67	**96.73 ± 3.37**	97.21 ± 2.55	93.73 ± 3.83
Ger	76.94 ± 1.02	**71.80 ± 3.58**	77.99 ± 2.19	67.20 ± 4.09	73.59 ± 2.59	66.30 ± 3.90	63.02 ± 2.11	63.90 ± 6.20	73.07 ± 0.83	69.80 ± 3.71
Gla	77.02 ± 1.88	69.60 ± 10.36	83.59 ± 4.73	65.71 ± 9.08	79.05 ± 4.56	65.21 ± 12.58	74.71 ± 1.26	**76.25 ± 10.08**	77.68 ± 1.76	70.85 ± 12.86
Hou	97.42 ± 0.58	94.62 ± 4.63	95.53 ± 5.16	93.62 ± 7.60	90.65 ± 2.83	89.64 ± 6.46	92.64 ± 2.50	92.18 ± 6.78	91.90 ± 2.80	92.40 ± 3.75
Iri	96.09 ± 0.94	**95.33 ± 5.33**	98.49 ± 7.82	94.67 ± 4.67	99.58 ± 0.84	93.33 ± 6.80	94.44 ± 0.50	94.67 ± 2.67	95.26 ± 0.82	94.67 ± 4.00
Mam	84.50 ± 0.57	**83.25 ± 5.19**	84.41 ± 2.53	75.03 ± 5.57	90.02 ± 1.85	75.34 ± 4.20	71.34 ± 2.09	71.39 ± 4.49	71.86 ± 6.66	69.72 ± 7.55
Pim	78.74 ± 0.98	72.27 ± 4.16	80.84 ± 3.05	**73.11 ± 3.40**	79.03 ± 1.97	69.32 ± 3.28	53.91 ± 8.48	67.85 ± 9.22	71.90 ± 0.49	67.83 ± 4.99
Son	97.15 ± 1.17	85.70 ± 5.99	89.70 ± 3.85	77.79 ± 11.10	85.77 ± 3.00	66.33 ± 12.05	88.09 ± 1.47	86.02 ± 10.19	93.22 ± 0.48	**91.83 ± 7.14**
Spe	90.25 ± 0.93	**78.66 ± 5.31**	79.89 ± 2.76	69.73 ± 13.17	73.28 ± 8.65	67.92 ± 13.97	76.49 ± 1.35	73.53 ± 10.68	82.52 ± 0.61	74.53 ± 7.76
Tic	85.48 ± 1.30	**83.51 ± 5.95**	92.35 ± 5.36	69.31 ± 8.00	94.84 ± 0.93	72.97 ± 2.75	65.36 ± 0.15	65.35 ± 1.32	70.40 ± 0.47	70.36 ± 2.53
Veh	74.97 ± 0.99	70.85 ± 3.35	83.49 ± 1.90	65.99 ± 4.15	82.55 ± 2.15	63.36 ± 5.17	71.04 ± 1.11	69.85 ± 3.25	74.43 ± 0.54	**71.64 ± 4.26**
Wis	97.82 ± 0.31	**96.09 ± 2.15**	96.78 ± 11.13	95.13 ± 5.35	95.16 ± 5.80	92.70 ± 15.70	95.44 ± 0.41	95.71 ± 2.78	96.55 ± 0.54	95.28 ± 2.30
Zoo	98.50 ± 1.76	**94.97 ± 5.22**	98.13 ± 5.74	92.64 ± 7.70	99.79 ± 0.64	93.22 ± 5.69	93.91 ± 2.06	91.97 ± 7.16	89.16 ± 6.22	87.50 ± 11.01
Avg.	86.45 ± 1.22	**81.64 ± 5.59**	88.06 ± 4.13	75.53 ± 6.46	87.54 ± 2.98	73.17 ± 7.07	74.56 ± 2.13	74.72 ± 6.28	80.71 ± 2.04	77.53 ± 6.28

Table 27
IFS-CoCo vs Classical algorithms (Kappa in training and test phases).

Alg	IFS-CoCo		DROP3		ICF		Relief		LVW	
	Training	Test	Training	Test	Training	Test	Training	Test	Training	Test
Aut	83.10 ± 1.72	69.75 ± 8.95	88.52 ± 1.54	51.57 ± 8.51	89.66 ± 2.53	44.72 ± 6.99	70.51 ± 1.74	71.76 ± 8.54	78.94 ± 3.90	**71.95 ± 9.71**
Bal	74.94 ± 1.48	**75.78 ± 5.12**	68.17 ± 3.64	66.21 ± 3.72	55.90 ± 1.86	44.97 ± 5.62	20.08 ± 1.23	20.23 ± 5.10	63.08 ± 0.87	63.51 ± 7.06
Bup	51.19 ± 3.53	**31.15 ± 10.32**	55.35 ± 3.90	16.57 ± 6.85	33.80 ± 3.99	4.97 ± 9.77	9.35 ± 2.23	3.64 ± 10.38	27.87 ± 1.38	20.33 ± 9.65
Car	79.89 ± 0.95	**77.87 ± 2.13**	87.23 ± 2.14	34.85 ± 4.40	94.58 ± 1.15	58.15 ± 3.54	11.62 ± 1.32	10.31 ± 1.02	54.57 ± 0.28	54.25 ± 1.43
Cle	35.83 ± 2.03	**30.43 ± 9.02**	77.32 ± 3.46	21.45 ± 4.24	68.97 ± 6.16	22.25 ± 5.93	11.00 ± 7.84	9.54 ± 9.23	34.47 ± 5.50	16.09 ± 9.85
Der	98.48 ± 0.43	91.82 ± 3.92	89.87 ± 3.56	91.14 ± 3.90	84.99 ± 1.93	85.78 ± 3.91	95.86 ± 0.77	**95.90 ± 3.07**	96.50 ± 2.55	92.15 ± 3.63
Ger	38.30 ± 0.82	23.35 ± 3.38	54.10 ± 2.29	22.10 ± 3.99	43.28 ± 2.59	24.07 ± 3.60	11.76 ± 2.21	13.31 ± 6.50	34.49 ± 0.83	**27.79 ± 3.61**
Gla	69.01 ± 1.28	55.44 ± 10.12	77.79 ± 4.83	53.01 ± 8.78	72.64 ± 4.56	46.88 ± 12.68	65.53 ± 1.36	**67.48 ± 9.78**	69.84 ± 1.76	60.87 ± 12.56
Hou	94.48 ± 0.68	**87.00 ± 4.43**	62.33 ± 5.16	72.54 ± 7.70	80.25 ± 2.93	78.94 ± 6.56	84.46 ± 2.40	83.75 ± 6.58	82.80 ± 2.80	83.81 ± 3.55
Iri	96.89 ± 1.04	**94.37 ± 4.83**	95.67 ± 7.92	93.00 ± 4.67	96.13 ± 0.94	92.50 ± 6.60	91.67 ± 0.40	92.00 ± 2.97	92.89 ± 0.82	92.00 ± 4.20
Mam	68.87 ± 0.66	**65.83 ± 6.29**	68.79 ± 2.43	49.57 ± 5.37	80.00 ± 1.95	50.30 ± 4.10	42.15 ± 1.99	42.35 ± 4.29	43.60 ± 6.66	39.22 ± 7.25
Pim	50.53 ± 0.96	**36.18 ± 4.36**	61.48 ± 2.95	32.96 ± 3.20	57.08 ± 2.07	26.69 ± 3.58	9.64 ± 8.58	10.99 ± 9.42	38.11 ± 0.49	28.85 ± 4.99
Son	94.19 ± 1.07	73.76 ± 5.31	79.30 ± 3.75	55.09 ± 11.00	70.47 ± 3.00	34.07 ± 11.85	75.94 ± 1.57	71.57 ± 10.09	86.30 ± 0.48	**83.46 ± 6.84**
Spe	70.36 ± 0.83	26.09 ± 5.79	51.41 ± 2.76	17.91 ± 13.17	45.32 ± 8.75	5.79 ± 14.27	31.68 ± 1.35	27.16 ± 10.58	47.49 ± 0.61	**28.80 ± 7.46**
Tic	69.33 ± 1.30	**60.02 ± 5.69**	72.87 ± 5.26	21.00 ± 8.10	83.25 ± 0.93	27.94 ± 2.75	37.46 ± 0.05	15.48 ± 1.32	39.36 ± 0.47	19.09 ± 2.43
Veh	66.85 ± 0.91	60.88 ± 3.78	77.51 ± 1.90	47.95 ± 4.05	76.68 ± 2.05	51.11 ± 5.07	61.37 ± 1.01	59.79 ± 3.05	65.89 ± 0.54	**62.16 ± 4.16**
Wis	94.40 ± 0.31	**94.37 ± 2.70**	96.61 ± 11.23	89.34 ± 5.15	93.27 ± 5.80	88.53 ± 15.50	89.86 ± 0.51	90.49 ± 2.68	92.37 ± 0.54	89.53 ± 2.50
Zoo	95.16 ± 1.56	**90.82 ± 4.94**	92.18 ± 5.64	90.27 ± 8.00	93.25 ± 0.74	90.74 ± 5.79	86.67 ± 1.96	89.41 ± 7.06	85.76 ± 6.22	83.40 ± 11.31
Avg.	73.99 ± 1.20	**63.61 ± 5.62**	75.36 ± 4.13	51.47 ± 6.38	73.31 ± 3.00	48.80 ± 7.12	50.37 ± 2.14	48.62 ± 6.20	63.02 ± 2.04	56.52 ± 6.23

Table 28
Reduction rates achieved (Classical methods).

Alg	IFS-CoCo (IS)	IFS-CoCo (FS)	DROP3	ICF	Relief	LVW
Aut	99.62	69.20	57.57	51.60	19.99	47.20
Bal	97.51	65.00	86.76	93.96	25.00	0.00
Bup	97.97	38.33	70.05	70.46	51.66	28.33
Car	95.18	16.67	88.34	85.55	46.67	16.67
Cle	98.02	54.62	83.17	79.72	72.31	48.46
Der	99.88	55.88	92.32	70.34	18.24	45.59
Ger	97.39	43.00	78.36	73.60	71.50	42.01
Gla	98.60	41.11	74.15	67.75	17.78	44.44
Hou	99.11	66.25	93.00	87.51	35.62	63.75
Iri	95.93	57.50	92.30	64.22	10.00	19.99
Mam	98.53	78.00	82.09	57.69	54.00	26.00
Pim	98.38	65.00	82.13	77.29	77.50	46.25
Son	99.36	57.17	75.91	71.21	58.50	52.17
Spe	98.13	59.09	83.39	89.22	44.55	52.95
Tic	98.61	24.44	92.87	70.90	54.44	22.22
Veh	96.98	42.78	77.27	69.16	43.33	44.99
Wis	99.38	45.56	97.47	95.28	3.33	34.44
Zoo	98.72	56.32	83.51	43.60	16.25	31.25
Avg.	98.18	52.00	82.81	73.28	40.04	37.04

Table 29
Time elapsed (Classical methods).

Alg	IFS-CoCo	DROP3	ICF	Relief	LVW
Aut	17.76	0.06	0.02	0.07	27.34
Bal	69.58	0.28	0.05	0.21	58.56
Bup	24.27	0.05	0.03	0.05	26.34
Car	884.80	2.83	0.53	2.64	585.21
Cle	31.87	0.05	0.03	0.09	30.83
Der	93.83	0.24	0.08	0.17	118.39
Ger	555.26	0.63	0.28	0.54	519.59
Gla	14.89	0.05	0.02	0.06	12.12
Hou	74.04	0.38	0.08	0.26	81.76
Iri	5.46	0.06	0.02	0.04	3.55
Mam	223.67	0.59	0.19	0.47	183.84
Pim	197.61	0.28	0.11	0.23	146.32
Son	44.05	0.08	0.03	0.06	102.97
Spe	57.05	0.09	0.05	0.07	138.40
Tic	304.43	0.66	0.17	0.73	255.18
Veh	300.99	0.47	0.20	1.05	273.94
Wis	180.22	1.09	0.13	0.92	140.83
Zoo	3.88	0.03	0.02	0.03	4.65
Avg.	171.31	0.44	0.11	0.43	150.55

Table 30
IFS-CoCo vs IS algorithms (Accuracy in training and test phases, high size data sets).

Alg	IFS-CoCo		IS-CHC		IS-SSGA		IS-GGA		1-NN	
	Training	Test	Training	Test	Training	Test	Training	Test	Training	Test
Che	96.91 ± 0.60	$\mathbf{96.56 \pm 2.45}$	87.81 ± 0.61	85.27 ± 4.45	92.04 ± 0.61	85.00 ± 3.15	88.77 ± 0.59	86.56 ± 2.55	84.56 ± 0.35	84.70 ± 2.65
Mov	90.52 ± 1.69	$\mathbf{86.66 \pm 7.34}$	71.67 ± 1.68	65.00 ± 7.37	83.33 ± 1.69	63.89 ± 7.35	79.32 ± 1.69	63.89 ± 7.35	81.48 ± 1.42	81.94 ± 7.35
Sat	91.51 ± 0.49	$\mathbf{91.29 \pm 0.97}$	88.07 ± 0.47	87.87 ± 0.95	91.56 ± 0.47	89.74 ± 0.98	90.35 ± 0.49	90.98 ± 0.98	90.85 ± 0.41	90.58 ± 0.99
Spa	93.68 ± 0.46	$\mathbf{93.69 \pm 2.19}$	87.99 ± 0.48	88.48 ± 2.19	91.23 ± 0.50	86.96 ± 2.20	89.85 ± 0.47	88.91 ± 2.19	89.99 ± 0.43	89.45 ± 2.23
Spl	87.70 ± 0.54	$\mathbf{86.83 \pm 1.30}$	73.63 ± 0.59	71.16 ± 1.33	82.34 ± 0.56	73.35 ± 1.33	79.03 ± 0.55	76.80 ± 1.30	75.24 ± 0.50	74.95 ± 1.33
Tex	98.99 ± 0.57	98.36 ± 1.46	94.00 ± 0.55	93.82 ± 1.47	98.14 ± 0.57	95.45 ± 1.46	97.29 ± 0.56	95.09 ± 1.47	99.01 ± 0.52	$\mathbf{99.05 \pm 1.47}$
Avg.	93.22 ± 0.73	$\mathbf{92.23 \pm 2.62}$	83.86 ± 0.73	81.93 ± 2.96	89.77 ± 0.73	82.40 ± 2.75	87.44 ± 0.73	83.71 ± 2.64	87.93 ± 0.61	86.78 ± 2.67

Table 31
IFS-CoCo vs FS algorithms (Accuracy in training and test phases, high size data sets).

Alg	IFS-CoCo		FS-CHC		FS-SSGA		FS-GGA		1-NN	
	Training	Test	Training	Test	Training	Test	Training	Test	Training	Test
Che	96.91 ± 0.60	$\mathbf{96.56 \pm 2.45}$	98.61 ± 0.57	95.43 ± 2.45	78.48 ± 0.57	82.50 ± 2.65	78.58 ± 0.58	81.56 ± 3.85	84.56 ± 0.35	84.70 ± 2.65
Mov	90.52 ± 1.69	$\mathbf{86.66 \pm 7.34}$	92.59 ± 1.69	80.56 ± 7.38	87.65 ± 1.65	77.78 ± 7.37	87.04 ± 1.65	77.78 ± 7.35	81.48 ± 1.42	81.94 ± 7.35
Sat	91.51 ± 0.49	$\mathbf{91.29 \pm 0.97}$	91.70 ± 0.49	91.07 ± 0.97	91.66 ± 0.50	90.20 ± 0.97	91.70 ± 0.47	90.36 ± 0.96	90.85 ± 0.41	90.58 ± 0.99
Spa	93.68 ± 0.46	$\mathbf{93.69 \pm 2.19}$	93.52 ± 0.50	92.39 ± 2.19	91.95 ± 0.48	91.74 ± 2.19	92.02 ± 0.47	92.39 ± 2.22	89.99 ± 0.43	89.45 ± 2.23
Spl	87.70 ± 0.54	$\mathbf{86.83 \pm 1.30}$	86.52 ± 0.57	80.56 ± 1.33	88.40 ± 0.57	84.95 ± 1.31	87.22 ± 0.57	84.01 ± 1.34	75.24 ± 0.50	74.95 ± 1.33
Tex	98.99 ± 0.57	98.36 ± 1.46	99.41 ± 0.56	98.55 ± 1.46	99.31 ± 0.58	98.11 ± 1.47	99.43 ± 0.58	98.34 ± 1.50	99.01 ± 0.52	$\mathbf{99.05 \pm 1.47}$
Avg.	93.22 ± 0.73	$\mathbf{92.23 \pm 2.62}$	93.73 ± 0.73	89.76 ± 2.63	89.58 ± 0.72	87.55 ± 2.66	89.33 ± 0.72	87.41 ± 2.87	87.93 ± 0.61	86.78 ± 2.67

Table 32
IFS-CoCo vs IFS algorithms (Accuracy in training and test phases, high size data sets).

Alg	IFS-CoCo		IFS-CHC		IFS-IGA		IFS-HGA		1-NN	
	Training	Test	Training	Test	Training	Test	Training	Test	Training	Test
Che	96.91 ± 0.60	$\mathbf{96.56 \pm 2.45}$	94.32 ± 0.58	94.34 ± 1.45	88.57 ± 0.59	88.37 ± 2.85	93.22 ± 0.61	91.22 ± 2.85	84.56 ± 0.35	84.70 ± 2.65
Mov	90.52 ± 1.69	$\mathbf{86.66 \pm 7.34}$	70.00 ± 1.67	65.83 ± 7.35	79.34 ± 1.68	72.34 ± 7.38	80.21 ± 1.67	70.21 ± 7.37	81.48 ± 1.42	81.94 ± 7.35
Sat	91.51 ± 0.49	$\mathbf{91.29 \pm 0.97}$	87.34 ± 0.48	86.11 ± 0.97	85.83 ± 0.49	83.83 ± 0.97	89.70 ± 0.48	85.85 ± 0.97	90.85 ± 0.41	90.58 ± 0.99
Spa	93.68 ± 0.46	$\mathbf{93.69 \pm 2.19}$	91.37 ± 0.47	90.71 ± 2.20	91.92 ± 0.46	91.12 ± 2.20	92.75 ± 0.46	91.56 ± 2.20	89.99 ± 0.43	89.45 ± 2.23
Spl	87.70 ± 0.54	$\mathbf{86.83 \pm 1.30}$	88.37 ± 0.56	86.06 ± 1.32	79.15 ± 0.56	78.65 ± 1.32	86.54 ± 0.59	83.54 ± 1.34	75.24 ± 0.50	74.95 ± 1.33
Tex	98.99 ± 0.57	98.36 ± 1.46	93.57 ± 0.57	93.24 ± 1.47	92.98 ± 0.59	92.21 ± 1.46	98.32 ± 0.56	95.52 ± 1.49	99.01 ± 0.52	$\mathbf{99.05 \pm 1.47}$
Avg.	93.22 ± 0.73	$\mathbf{92.23 \pm 2.62}$	87.49 ± 0.72	86.05 ± 2.46	86.30 ± 0.73	84.42 ± 2.70	90.12 ± 0.73	86.32 ± 2.70	87.93 ± 0.61	86.78 ± 2.67

Table 33
Reduction rates achieved (high size data sets).

Alg	IFS-CoCo (IS)	IS-CHC	IS-SSGA	IS-GGA	IFS-CoCo (FS)	FS-CHC	FS-SSGA	FS-GGA	IFS-CoCo (FS)	IFS-CHC	IFS-IGA	IFS-HGA
Che	81.64	98.19	94.95	94.23	38.89	33.33	34.12	33.89	99.64	99.69	99.72	42.12
Mov	99.33	84.88	84.58	83.22	63.33	64.44	65.76	65.98	98.89	89.20	92.33	72.81
Sat	71.65	99.36	97.99	98.14	47.22	30.56	31.23	31.45	99.12	99.55	99.58	45.90
Spa	77.16	99.20	98.12	98.22	49.12	54.39	56.53	55.98	99.02	99.44	99.49	63.31
Spl	92.96	99.02	95.42	95.65	76.67	71.67	71.98	71.89	99.12	98.61	98.82	82.42
Tex	79.56	97.01	96.18	96.95	65.00	60.00	62.50	62.50	98.15	98.55	98.14	69.45
Avg.	83.72	96.28	94.54	94.40	56.71	52.40	53.69	53.62	98.99	97.51	98.01	62.67

Table 34
IFS-CoCo vs IS algorithms (Kappa in training and test phases, high size data sets).

Alg	IFS-CoCo		IS-CHC		IS-SSGA		IS-GGA		1-NN	
	Training	Test	Training	Test	Training	Test	Training	Test	Training	Test
Che	93.79 ± 0.74	$\mathbf{93.09 \pm 3.23}$	66.73 ± 0.56	67.38 ± 2.62	84.01 ± 0.68	69.79 ± 1.93	77.45 ± 0.63	73.08 ± 3.39	84.56 ± 1.09	67.68 ± 2.22
Mov	91.40 ± 1.56	$\mathbf{79.16 \pm 7.82}$	72.54 ± 2.24	55.45 ± 7.56	82.14 ± 1.95	61.29 ± 7.46	77.84 ± 2.26	61.29 ± 7.46	81.48 ± 1.81	76.20 ± 7.80
Sat	89.51 ± 0.80	$\mathbf{89.26 \pm 0.91}$	85.25 ± 0.81	84.99 ± 1.30	89.55 ± 0.31	87.31 ± 1.07	88.05 ± 0.94	88.82 ± 1.14	90.85 ± 0.42	88.30 ± 1.27
Spa	85.16 ± 1.04	$\mathbf{86.68 \pm 2.33}$	74.33 ± 0.81	74.93 ± 2.01	81.42 ± 0.51	72.14 ± 2.08	78.63 ± 0.54	76.53 ± 2.64	89.99 ± 0.50	78.97 ± 2.67
Spl	80.39 ± 0.93	$\mathbf{79.18 \pm 1.20}$	56.09 ± 1.10	50.83 ± 1.87	71.58 ± 0.94	57.92 ± 1.73	66.35 ± 0.55	63.06 ± 1.76	75.24 ± 0.45	61.37 ± 1.30
Tex	98.89 ± 1.06	$\mathbf{98.80 \pm 1.49}$	93.40 ± 0.65	93.20 ± 1.54	97.96 ± 1.04	95.00 ± 1.30	97.02 ± 1.16	94.60 ± 1.52	99.01 ± 1.03	98.60 ± 1.96
Avg.	89.85 ± 1.02	$\mathbf{87.69 \pm 2.83}$	74.72 ± 1.03	71.13 ± 2.82	84.44 ± 0.90	73.91 ± 2.60	80.89 ± 1.01	76.23 ± 2.99	79.43 ± 0.88	78.52 ± 2.87

Table 35
IFS-CoCo vs FS algorithms (Kappa in training and test phases, high size data sets).

Alg	IFS-CoCo		FS-CHC		FS-SSGA		FS-GGA		1-NN	
	Training	Test	Training	Test	Training	Test	Training	Test	Training	Test
Che	93.79 ± 0.74	$\mathbf{93.09 \pm 3.23}$	97.21 ± 1.09	92.35 ± 3.29	56.14 ± 1.09	64.49 ± 3.12	56.38 ± 1.00	62.55 ± 1.92	84.56 ± 1.09	67.68 ± 2.22
Mov	91.40 ± 1.56	$\mathbf{79.16 \pm 7.82}$	92.06 ± 2.08	79.11 ± 7.59	86.77 ± 1.57	76.20 ± 7.20	86.11 ± 1.53	76.20 ± 7.18	81.48 ± 1.81	76.20 ± 7.80
Sat	89.51 ± 0.80	$\mathbf{89.26 \pm 0.91}$	89.75 ± 0.73	88.22 ± 1.15	89.71 ± 0.38	87.93 ± 0.97	89.75 ± 0.73	88.14 ± 1.53	90.85 ± 0.42	88.30 ± 1.27
Spa	85.16 ± 1.04	$\mathbf{86.68 \pm 2.33}$	86.51 ± 0.93	84.22 ± 2.54	83.13 ± 0.72	82.73 ± 2.35	83.34 ± 0.62	84.13 ± 2.43	89.99 ± 0.50	78.97 ± 2.67
Spl	80.39 ± 0.93	$\mathbf{79.18 \pm 1.20}$	78.52 ± 0.73	69.56 ± 1.21	81.50 ± 0.97	76.40 ± 1.12	79.60 ± 0.59	74.65 ± 1.87	75.24 ± 0.45	61.37 ± 1.30
Tex	98.89 ± 1.06	98.80 ± 1.49	99.36 ± 1.01	$\mathbf{98.83 \pm 1.59}$	99.24 ± 1.17	98.50 ± 1.90	99.38 ± 0.46	98.20 ± 1.59	99.01 ± 1.03	98.60 ± 1.96
Avg.	89.85 ± 1.02	$\mathbf{87.69 \pm 2.83}$	90.57 ± 1.10	85.38 ± 2.90	82.75 ± 0.98	81.04 ± 2.78	82.43 ± 0.82	80.64 ± 2.75	79.43 ± 0.88	78.52 ± 2.87

Table 36
IFS-CoCo vs IFS algorithms (Kappa in training and test phases, high size data sets).

Alg	IFS-CoCo		IFS-CHC		IFS-IGA		IFS-HGA		1-NN	
	Training	Test	Training	Test	Training	Test	Training	Test	Training	Test
Che	93.79 ± 0.74	$\mathbf{93.09 \pm 3.23}$	87.92 ± 0.47	89.95 ± 2.73	80.92 ± 0.56	82.95 ± 3.17	91.92 ± 0.64	88.15 ± 3.01	84.56 ± 1.09	67.68 ± 2.22
Mov	91.40 ± 1.56	$\mathbf{79.16 \pm 7.82}$	67.25 ± 2.09	52.44 ± 7.56	73.48 ± 1.65	65.84 ± 7.44	87.01 ± 1.53	73.44 ± 7.17	81.48 ± 1.81	76.20 ± 7.80
Sat	89.51 ± 0.80	$\mathbf{89.26 \pm 0.91}$	83.87 ± 1.04	82.20 ± 1.12	81.02 ± 0.34	80.02 ± 1.31	85.92 ± 0.92	85.52 ± 1.56	90.85 ± 0.42	88.30 ± 1.27
Spa	85.16 ± 1.04	$\mathbf{86.68 \pm 2.33}$	82.20 ± 0.96	85.92 ± 2.72	78.15 ± 0.30	85.67 ± 2.36	82.90 ± 0.99	86.01 ± 2.36	89.99 ± 0.50	78.97 ± 2.67
Spl	80.39 ± 0.93	$\mathbf{79.18 \pm 1.20}$	80.25 ± 0.45	76.49 ± 1.22	67.25 ± 0.65	57.49 ± 1.21	76.25 ± 0.82	72.49 ± 1.11	75.24 ± 0.45	61.37 ± 1.30
Tex	98.89 ± 1.06	$\mathbf{98.80 \pm 1.49}$	93.24 ± 0.38	90.00 ± 1.29	92.88 ± 0.73	88.14 ± 1.76	97.17 ± 0.82	98.18 ± 2.02	99.01 ± 1.03	98.60 ± 1.96
Avg.	89.85 ± 1.02	$\mathbf{87.69 \pm 2.83}$	82.46 ± 0.90	79.50 ± 2.77	78.95 ± 0.71	76.69 ± 2.88	86.86 ± 0.95	83.97 ± 2.87	79.43 ± 0.88	78.52 ± 2.87

Table 37
Time elapsed (high size data sets).

Alg	IFS-CoCo	IS-CHC	IS-SSGA	IS-GGA	FS-CHC	FS-SSGA	FS-GGA	IFS-CHC	IFS-IGA	IFS-HGA
Che	13068	770	1632	1732	21932	20657	20946	528	8841	6549
Mov	173	34	91	99	242	221	226	31	94	86
Sat	87094	2930	9275	10112	133700	125678	129634	3011	45672	42135
Spa	54636	1816	5430	6002	79048	74563	74579	1948	29220	27844
Spl	22218	677	1646	1843	35081	33264	32765	848	14794	10986
Tex	68297	2542	7689	7902	76481	74533	74987	2283	36557	32895
Avg.	40914	1461	4294	4615	57747	54819	55523	1442	22530	20082

rates achieved. Table 17 shows the running times of every method.

In a similar way, Tables 18–21 show the results obtained by FS methods, and Tables 22, 23, 24 and 25 show the results obtained by IFS methods. Finally, Tables 26–29 show the full results of the comparison between IFS-CoCo and the classical approaches of IS and FS.

For the second study, Tables 30–32 shows the average accuracy results obtained over the 6 high dimensional data sets. Table 33 shows the average reduction rates achieved over these domains. Tables 34–36 shows the average kappa results, and finally, Table 37 shows the average running times obtained.

Note: For space reasons, accuracy and kappa results are shown in the format $xx.xx \pm x.xx$ instead of $0.xxxx \pm 0.xxxx$.

References

[1] I.H. Witten, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques, second ed., Morgan Kaufmann, Los Altos, CA, 2005.
[2] X. Wu, V. Kumar (Eds.), The Top Ten Algorithms in Data Mining, Chapman & Hall, CRC, London, Boca Raton, 2009.
[3] D. Pyle, Data Preparation for Data Mining, Morgan Kaufmann, San Francisco, 1999.
[4] S. Wang-Manoranjan, D.C. Xu, Efficient data reduction in multimedia data, Applied Intelligence 25 (2006) 359–374.
[5] A. Kolesnikov, P. Frantib, Data reduction of large vector graphics, Pattern Recognition 38 (2005) 381–394.
[6] S.W. Kim, B.J. Oomenn, On using prototype reduction schemes to optimize dissimilarity-based classification, Pattern Recognition 40 (11) (2007) 2946–2957.
[7] J.R. Cano, S. García, F. Herrera, Subgroup discovery in large size data sets preprocessed using stratified instance selection for increasing the presence of minority classes, Pattern Recognition Letters 29 (2008) 2156–2164.
[8] S.W. Kim, B.J. Oomenn, On using prototype reduction schemes to enhance the computation of volume-based inter-class overlap measures, Pattern Recognition 42(11) (2009) 2695–2704.
[9] T.M. Cover, P.E. Hart, Nearest neighbor pattern classification, IEEE Transactions on Information Theory 13 (1967) 21–27.
[10] P. Perner, Prototype-based classification, Applied Intelligence 28 (2008) 238–246.
[11] H. Liu, H. Motoda (Eds.), Instance Selection and Construction for Data Mining, Springer, New York, 2001.
[12] H. Liu, H. Motoda (Eds.), Computational Methods of Feature Selection, Chapman & Hall, CRC, London, Boca Raton, 2007.
[13] J.R. Cano, F. Herrera, M. Lozano, Using evolutionary algorithms as instance selection for data reduction in KDD: An experimental study, IEEE Transactions on Evolutionary Computation 7 (2003) 561–575.
[14] A.E. Eiben, J.E. Smith, Introduction to Evolutionary Computing, Springer, Berlin, 2003.
[15] S. García, J.R. Cano, F. Herrera, A memetic algorithm for evolutionary prototype selection: a scaling up approach, Pattern Recognition 41 (8) (2008) 2693–2709.
[16] L.I. Kuncheva, Editing for the *k*-nearest neighbors rule by a genetic algorithm, Pattern Recognition Letters 16 (1995) 809–814.
[17] I. Inza, P. Larraaga, B. Sierra, Feature subset selection by Bayesian networks: a comparison with genetic and sequential algorithms, International Journal of Approximate Reasoning 27 (2001) 143–164.
[18] I. Oh, J. Lee, B. Moon, Hybrid Genetic Algorithms for Feature Selection, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (2004) 1424–1437.
[19] D. Whitley, C. Guerra-Salcedo, Genetic search for feature subset selection: a comparison between CHC and GENESIS, in: Proceedings of the Third Annual Conference on Genetic Programming, Wisconsin, 1998, pp. 504–509.
[20] A.A. Freitas, Data Mining and Knowledge Discovery with Evolutionary Algorithms, Springer, New York, 2002.
[21] A. Ghosh, L.C. Jain, Evolutionary Computation in Data Mining, Springer, Berlin, 2005.
[22] S. Bandyopadhyay, S. Santanu, A genetic approach for efficient outlier detection in projected space, Pattern Recognition 41 (2008) 1338–1349.
[23] M.A. Potter, K.A. De Jong, Cooperative coevolution: an architecture for evolving coadapted subcomponents, Evolutionary Computation 8 (2000) 1–29.
[24] D.H. Wolpert, W.G. Macready, Coevolutionary free lunches, IEEE Transactions on Evolutionary Computation 9 (2005) 721–735.
[25] N. Garcia-Pedrajas, D. Ortiz-Boyer, A cooperative constructive method for neural networks for pattern recognition, Pattern Recognition 40 (1) (2007) 80–98.
[26] R.P. Wiegand, T. Jansen, The cooperative coevolutionary (1+1) EA, Evolutionary Computation 12 (2004) 405–434.
[27] F. Wilcoxon, Individual comparisons by rankings methods, Biometrics 1 (1945) 80–83.
[28] H. Liu, H. Motoda, On issues of instance selection, Data Mining and Knowledge Discovery 6 (2) (2002) 115–130.
[29] J.R. Cano, F. Herrera, M. Lozano, Evolutionary stratified training set selection for extracting classification rules with trade-off precision-interpretability, Data and Knowledge Engineering 60 (2007) 90–100.
[30] K. Kim, Artificial neural networks with evolutionary instance selection for financial forecasting, Expert Systems with Applications 30 (2006) 519–526.
[31] D.R. Wilson, T.R. Martinez, Reduction techniques for instance-based learning algorithms, Machine Learning 38 (2000) 257–286.
[32] P.E. Hart, The condensed nearest neighbor rule, IEEE Transactions on Information Theory 18 (1968) 515–516.
[33] D.L. Wilson, Asymptotic properties of nearest neighbor rules using edited data, IEEE Transactions on Systems, Man and Cybernetics 3 (1972) 408–421.
[34] E. Marchiori, Hit miss networks with applications to instance selection, Journal of Machine Learning Research 9 (2008) 997–1017.
[35] J.A. Olvera-López, J.A. Carrasco-Ochoa, J.F. Martínez-Trinidad, A new fast prototype selection method based on clustering, Pattern Analysis and Applications (2009), in press, doi:10.1007/s10044-008-0142-x.
[36] J.C. Bezdek, L.I. Kuncheva, Nearest prototype classifier designs: an experimental study, International Journal of Intelligent Systems 16 (2001) 1445–1473.
[37] N. Jankowski, M. Grochowski, Comparison of instances selection algorithms I. Algorithms survey, in: Lecture Notes in Computer Science, vol. 3070, Springer, Berlin, 2004, pp. 598–603.
[38] S.W. Kim, B.J. Oomenn, A brief taxonomy and ranking of creative prototype reduction schemes, Pattern Analysis and Applications 6 (2003) 232–244.
[39] R. Kohavi, G. John, Wrappers for feature selection, Artificial Intelligence 97 (1997) 273–324.
[40] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, Journal of Machine Learning Research 3 (2003) 1157–1182.
[41] Y. Saeys, I. Inza, P. Larranaga, A review of feature selection techniques in bioinformatics, Bioinformatics 19 (2007) 2507–2517.
[42] H. Liu, H. Motoda, Feature Selection for Knowledge Discovery and Data Mining, Springer, New York, 1998.
[43] Y. Li, B.L. Lu, Feature selection based on loss-margin of nearest neighbor classification, Pattern Recognition 42 (9) (2009) 1914–1921.
[44] D.J. Stracuzzi, P.E. Utgoff, Randomized variable elimination, Journal of Machine Learning Research 5 (2004) 1331–1362.
[45] J. Shie, S. Chen, Feature subset selection based on fuzzy entropy measures for handling classification problems, Applied Intelligence 28 (2008) 69–82.
[46] H. Liu, L. Yu, Toward integrating feature selection algorithms for classification and clustering, IEEE Transactions on Knowledge and Data Engineering 17 (3) (2005) 1–12.
[47] L.I. Kuncheva, L.C. Jain, Nearest neighbor classifier: simultaneous editing and descriptor selection, Pattern Recognition Letters 20 (1999) 1149–1156.
[48] H. Ishibuchi, T. Nakashima, M. Nii, Genetic-algorithm-based instance and feature selection, in: H. Liu, H. Motoda (Eds.), Instance Selection and Construction for Data Mining, 2001, pp. 95–112.
[49] J. Teixeira, R.A. Ferreira, G.A. Lima, A novel approach for integrating feature and instance selection, in: International Conference on Machine Learning and Cybernetics, Kunming, 2008, pp. 374–379.
[50] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, Optimization by simulated annealing, Science 4598 (1983) 671–680.
[51] H. Ahn, K. Kim, Bankruptcy prediction modeling with hybrid case-based reasoning and genetic algorithms approach, Applied Soft Computing 9 (2009) 599–607.
[52] L.J. Eshelman, The CHC adaptative search algorithm: how to have safe search when engaging in nontraditional genetic recombination, in: G.J.E. Rawlins (Ed.), Foundations of Genetic Algorithms, , 1991, pp. 265–283.
[53] R. Gil-Pita, X. Yao, Evolving edited *k*-nearest neighbor classifiers, International Journal of Neural Systems 18 (6) (2008) 1–9.
[54] H. Ishibuchi, T. Nakashima, Evolution of reference sets in nearest neighbor classification, Lecture Notes in Computer Science vol. 1585 (1999) 82–89.
[55] B. Sierra, E. Lazkano, I. Inza, M. Merino, P. Larraaga, J. Quiroga, Prototype selection and feature subset selection by estimation of distribution algorithms. A case study in the survival of cirrhotic patients treated with TIPS, in: Lecture Notes in Artificial Intelligence, vol. 2101, Springer, Berlin, 2001, pp. 20–29.
[56] J. Bala, K.A. De Jong, J. Huang, H. Vafaie, H. Wechsler, Using learning to facilitate the evolution of features for recognizing visual concepts, Evolutionary Computation 4 (3) (1997) 297–311.
[57] J. Casillas, O. Cordon, M.J. Del Jesus, F. Herrera, Genetic feature selection in a fuzzy rule-based classification system learning process for high-dimensional problems, Information Sciences 136 (2001) 135–157.
[58] A. Gonzalez, R. Perez, Selection of relevant features in a fuzzy genetic learning algorithm, IEEE Transactions on Systems, Man and Cybernetics 31 (3) (2001) 417–425.
[59] L. Rokach, Genetic algorithm-based feature set partitioning for classification problems, Pattern Recognition 41 (2008) 1676–1700.
[60] W. Siedlecki, J. Sklansky, A note on genetic algorithm for large-scale feature selection, Pattern Recognition Letters 10 (1989) 335–347.
[61] C. Wang, Y. Huang, Evolutionary-based feature selection approaches with new criteria for data mining: a case study of credit approval data, Expert Systems with Applications 36 (2009) 5900–5908.
[62] P. Zhang, B. Verma, K. Kumar, Neural vs. statistical classifier in conjunction with genetic algorithm based feature selection, Pattern Recognition Letters 26 (7) (2005) 909–919.

[63] S. Ho, C. Liu, S. Liu, Design of an optimal nearest neighbor classifier using an intelligent genetic algorithm, Pattern Recognition Letters 23 (2002) 1495–1503.
[64] F. Ros, S. Guillaume, M. Pintore, J.R. Chretien, Hybrid genetic algorithm for dual selection, Pattern Analysis and Applications 11 (2008) 179–198.
[65] P.W. Price, Biological Evolution, Saunders College Publishing, 1998.
[66] R.P. Wiegand, An analysis of cooperative coevolutionary algorithms, Ph.D. Thesis, George Mason University, Fairfax, Virginia, 2003.
[67] C.D. Rosin, R.K. Belew, New Methods for competitive coevolution, Evolutionary Computation 15 (1997) 1–29.
[68] L. Panait, R.P. Wiegand, S. Luke, Improving coevolutionary search for optimal multiagent behaviors, in: International Joint Conferences on Artificial Intelligence, Acapulco, 2003, pp. 653–658.
[69] L. Panait, S. Luke, J.F Harrison, Archive-based cooperative coevolutionary algorithms, in: Genetic and Evolutionary Computation Conference, GECCO'06, Seattle, 2006, pp. 345–352.
[70] R.P. Wiegand, J. Sarma, Spatial embedding and loss of gradient in cooperative coevolutionary algorithms, Parallel Problem Solving from Nature VIII, Birmingham, 2004, pp. 912–921.
[71] E. Popovici, K.A. De Jong, Sequential versus parallel cooperative coevolutionary algorithms for optimization, IEEE Congress on Evolutionary Computation, Vancouver, 2006, pp. 1610–1617.
[72] R.P. Wiegand, L. Liles, K.A. De Jong, An empirical analysis of collaboration methods in cooperative coevolutionary algorithms, in: Genetic and Evolutionary Computation Conference, GECCO'01, San Francisco, 2001, pp. 1235–1242.
[73] J. Hofbauer, K. Sigmund, Evolutionary games and population dynamics, Cambridge University Press, Cambridge, 1998.
[74] D.H. Wolpert, W.G. Macready, No free lunch theorems for optimization, IEEE Transactions on Evolutionary Computation 1 (1) (1997) 67–82.
[75] C.S. Travis, D.R. Tauritz, A no-free-lunch framework for coevolution, in: Genetic and Evolutionary Computation Conference, GECCO'08, Atlanta, 2008, pp. 371–378.
[76] A. Asuncion, D.J. Newman, UCI repository of machine learning databases, 2007, URL: 〈 http://www.ics.uci.edu/∼mlearn/MLRepository.html 〉.
[77] E. Alpaydin, Introduction to Machine Learning, MIT Press, Cambridge, MA, 2004.
[78] T.S. Lim, W.Y. Loh, Y.S. Shih, A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms, Machine Learning 40 (3) (2000) 203–228.
[79] J. Cohen, A coefficient of agreement for nominal scales, Educational and Psychological Measurement 20 (1) (1960) 37–46.
[80] A. Ben-David, A lot of randomness is hiding in accuracy, Engineering Applications of Artificial Intelligence 20 (7) (2007) 875–885.
[81] J. Demšar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine Learning Research 7 (2006) 1–30.
[82] 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.
[83] D.J. Sheskin, Handbook of Parametric and Nonparametric Statistical Procedures, CRC Press, Boca Raton, 1997.
[84] J.H. Zar, Biostatistical Analysis, Prentice-Hall, Englewood Cliffs, London, 1999.
[85] H. Brighton, C. Mellish, Advances in instance selection for instance-based learning algorithms, Data Mining and Knowledge Discovery 6 (2) (2002) 153–172.
[86] K. Kira, L. Rendell, A practical approach to feature selection, in: P. Sleeman, P. Edwards (Eds.), Proceedings of the Ninth International Conference on Machine Learning (ICML-92), Morgan Kaufmann, Los Altos, CA, 1992, pp. 249–256.
[87] H. Liu, R. Setiono, Feature selection and classification: a probabilistic wrapper approach, in: Ninth International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, Fukuoka, Japan, 1996, pp. 419–424.
[88] T.K. Ho, M. Basu, Complexity measures of supervised classification problems, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (3) (2002) 289–300.
[89] S. Singh, Multiresolution estimates of classification complexity, IEEE Transactions on Pattern Analysis and Machine Intelligence 25 (12) (2003) 1534–1539.
[90] J.R. Cano, F. Herrera, M. Lozano, Stratification for scaling up evolutionary prototype selection, Pattern Recognition Letters 26 (2005) 953–963.
[91] A. Haro-García, N. García-Pedrajas, A divide-and-conquer recursive approach for scaling up instance selection algorithms, Data Mining and Knowledge Discovery 18 (2009) 392–418.
[92] R. Paredes, E. Vidal, Learning weighted metrics to minimize nearest-neighbor classification error, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (7) (2006) 1100–1110.

About the Author—JOAQUÍN DERRAC received the M.Sc. degree in computer science from the University of Granada, Granada, Spain, in 2008. He is currently a Ph.D. student in the Department of Computer Science and Artificial Intelligence, University of Granada, Granada, Spain.

His research interests include data mining, lazy learning and evolutionary algorithms.

About the Author—SALVADOR GARCÍA received the M.Sc. and Ph.D. degrees in computer science from the University of Granada, Granada, Spain, in 2004 and 2008, respectively. He is currently an Associate Professor in the Department of Computer Science, University of Jaén, Jaén, Spain.

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

About the Author—FRANCISCO HERRERA received the M.Sc. degree in Mathematics in 1988 and the Ph.D. degree in Mathematics in 1991, both from the University of Granada, Spain.

He is currently a Professor in the Department of Computer Science and Artificial Intelligence at the University of Granada. He has published more than 150 papers in international journals. He is coauthor of the book "Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases" (World Scientific, 2001). As edited activities, he has co-edited five international books and co-edited twenty special issues in international journals on different Soft Computing topics. He acts as associated editor of the journals: IEEE Transactions on Fuzzy Systems, Information Sciences, Mathware and Soft Computing, Advances in Fuzzy Systems, Advances in Computational Sciences and Technology, and International Journal of Applied Metaheuristic Computing. He currently serves as area editor of the Journal Soft Computing (area of genetic algorithms and genetic fuzzy systems), and he serves as member of several journal editorial boards, among others: Fuzzy Sets and Systems, Applied Intelligence, Knowledge and Information Systems, Information Fusion, Evolutionary Intelligence, International Journal of Hybrid Intelligent Systems, Memetic Computation.

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