Universidad de Granada

Departamento de Ciencias de la Computación
e Inteligencia Artificial

Programa de Doctorado en Ciencias de la Computación
y Tecnología Informática

Advanced Models for Nearest Neighbor Classification Based on Soft Computing Techniques

Tesis Doctoral

Joaquín Derrac Rus

Granada, Febrero de 2013

Editor: Editorial de la Universidad de Granada
Autor: Joaquín Derrac Rus
D.L.: GR 1886-2013
ISBN: 978-84-9028-608-1

Universidad de Granada

Advanced Models for Nearest Neighbor Classification Based on Soft Computing Techniques

MEMORIA PRESENTADA POR

Joaquín Derrac Rus

PARA OPTAR AL GRADO DE DOCTOR EN INFORMTICA

Febrero de 2013

DIRECTORES

Francisco Herrera Triguero y Salvador García López

Departamento de Ciencias de la Computación
e Inteligencia Artificial

La memoria titulada "*Advanced Models for Nearest Neighbor Classification Based on Soft Computing Techniques*", que presenta D. Joaquín Derrac Rus para optar al grado de doctor, ha sido realizada dentro del Programa Oficial de Doctorado en "*Ciencias de la Computación y Tecnología Informática*", en el Departamento de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Granada bajo la dirección de los doctores D. Francisco Herrera Triguero y D. Salvador García López.

El doctorando y los directores de la tesis garantizamos, al firmar esta tesis doctoral, que el trabajo ha sido realizado por el doctorando bajo la dirección de los directores de la tesis, y hasta donde nuestro conocimiento alcanza, en la realización del trabajo se han respetado los derechos de otros autores a ser citados cuando se han utilizado sus resultados o publicaciones.

Granada, Febrero de 2013

El Doctorando

Los directores

Fdo: Joaquín Derrac Rus

Fdo: Francisco Herrera Triguero

Fdo: Salvador García López

Tesis Doctoral Parcialmente Subvencionada por el Ministerio de Ciencia e Innovación bajo el Proyecto Nacional TIN2011-28488. También ha sido Subvencionada bajo el Programa de Becas-Contrato FPU del Plan Propio de la Universidad de Granada 2009, en su Resolución del 22 de Septiembre de 2009, y bajo el Programa de Becas de Formación de Profesorado Universitario del Ministerio de Educación, en su Resolución del 11 de Octubre de 2010, bajo la referencia AP2009-2923.

Agradecimientos

Quiero aprovechar estas líneas para expresar mi más profundo agradecimiento a todas aquellas personas que, con su apoyo, han hecho posible la realización de estas tesis. Tras recorrer este largo y duro camino, es reconfortante poder tomarse un momento de respiro, echar la vista atrás, y contemplar la larga lista de gente cuya mención en estas páginas es obligada.

En primer lugar quiero agradecer el esfuerzo, ánimo, trabajo, apoyo, tesón - y mil y un sustantivos más - de mis directores, Francisco Herrera y Salvador García. Desde los primeros momentos en que uno decide dedicarse a eso de *hacer la tesis* hasta los últimos compases en los que las correcciones finales, el papeleo y los plazos consumen los últimos instantes del doctorado, Paco y Salva han sido una guía fundamental a la hora de abordar todos los desafios de este periodo.

Después tengo que agradecer el papel de todos los mis compañeros de andanzas: A los *juniors*, Isaac, José Antonio, Jose y Christoph (sí, todavía os tengo que poner aquí), Michela, Benjamin, Álvaro, Rosa, Juan Antonio, Raquel, Daniel, Sergio, Sara, Pablo y Nele (*thank you so much!*); y a los ”no tan *juniors*”, Alberto, Julián, Salva (el mismo del párrafo de arriba, sí), Nacho, Manolo, Fran y Chris. También quiero expresar mi agradecimiento hacia el resto de miembros *senior* del grupo, y al rol que cada uno de vosotros habeis tenido como apoyo a la realizacin de esta tesis.

Este agradecimiento no sería completo si no incluyera también al resto de mis sufridos compañeros de despacho, tanto en *el dieciséis* - el lugar más íctico del departamento - como en el *DB5* del CITIC, nuestro flamante campo de batalla durante estos últimos años. *And I am also very grateful to Danny and Rebecca, for your invaluable assistance in our battle against this awkward, yet necessary language for everyday's work. Thank you!.*

I would also like to remember the support of the members of the DIGITS research group and the rest of people at the De Montfort University, and specially the inestimable help of Paco Chiclana during my research visit in Leicester. Thanks to them I have many remarkable experiences to tell from that exciting journey to the land where rain never ends.

Finalmente, pero no menos importante, quiero agradecer el papel, el cariño, la compresión y la paciencia de mis padres, Joaquín y María Angeles, y de mi hermana Elisabet durante todos estos años de mi vida (y sus piezas de viola, cuyo son ha acompañado muchos momentos especiales en estos años, como este mismo instante).

Y como no podía ser de otra manera, voy a concluir estos agradecimientos dedicándote las últimas palabras a tí, Victoria, por tu apoyo y por todo lo que has hecho por mí durante todos estos años. Al final parece que sí, que este camino tiene un final, y que lo tenemos mucho más cerca de lo que pensamos. !Ánimo!

GRACIAS A TODOS

Table of Contents

Chapter I

PhD dissertation

1. Introduction

In recent years, there has been a manifold increase in the amount of the data that applications in industry, medicine, economy and many other areas of research and business must manage. Researchers and practitioners in many application fields have an increasing necessity of new developments to gather, analyze and understand information which is continually stored in large data bases. The management and analysis of such a amount of data is beyond of human possibilities, and thus, it is necessary to rely in automatic procedures to acquire valuable knowledge, usually hidden in this contemporary *data deluge* [BHS09].

Therefore, the analysis of data and the extraction of useful knowledge has become a mandatory task - and a difficult challenge - in many business and research processes nowadays. This process, formally known as *Knowledge Discovery in Databases* (KDD) [FPSS96, Han05] represents the task of obtaining new knowledge from large amounts of data. These new pieces of knowledge should be useful and valid, and must make sense for the specific problem tackled in each scenario.

The KDD process is generally defined as a succession of five stages:

- Selection of the target data to which the KDD process will be applied.
- Preprocessing of the data, including several phases of cleaning and reduction to ease the procedures of the subsequent stages.
- Transforming the data into an structured representation; a representation as accurate as possible of the knowledge of the problem stored in the initial data.
- Construction of new knowledge patterns, through the analysis of the representations built in the former step.
- Visualization of the data, describing and understanding the new patterns obtained in a way in which they could be useful for the practitioners.

Following these stages, the KDD process includes a large number of computer science and artificial intelligence techniques and methodologies to perform numerous tasks in each step. Its central stages are also defined by the term of *data mining* [TSK05, WFH11]. Data mining is the discipline focused on the identification of patterns and the prediction of relationships from data. Usually, its techniques can be categorized as descriptive (when they are applied to discover interesting patterns among data) or predictive ones (when the behavior of a model is predicted through the analysis of the data available).

Both descriptive and predictive process of data mining are conducted by *machine learning* algorithms [MD01, Alp10, WFH11], introduced as mechanisms capable of induce knowledge from the available data. Such knowledge induction is desirable in problems with no efficient algorithmic solution, vaguely defined, or informally specified.

The use of machine learning algorithms is two-fold: They can be used just as black boxes, obtaining as a result only the output of the models. However, some of them can also be used as a tool of knowledge representation, building a symbolic knowledge structure aiming to be useful from the point of view of the functionality, but also from the perspective of interpretability.

Depending on their objectives, machine learning algorithms can be classified on two different areas:

- **Supervised learning**: Algorithms focused on using a set of labeled examples, describing information of several input variables, to predict the values of some output variables. The most common categories of supervised learning algorithms are classification (where the variable to predict is discrete; for example red, green, blue) and regression (where the variable to predict is continuous; for example: temperature, weight, ...).

- **Unsupervised learning**: Algorithms focused on searching for new patterns and relationships over unlabeled data. The most common categories of unsupervised learning algorithms are clustering (the process of splitting the data into several groups, with the examples belonging to each group being as similar as possible among them) and association (the identification of relationships from transactional data).

Classification methods can be defined as techniques that enable to learn how to categorize elements into several predefined classes. A classifier receives a data set as input, denoted as *training set*, and learns the classification model with it. In the validation process of the classifier, an extra set of examples, not used in the learning process (the *test set*) is used in order to check the accuracy of the classifier.

Generally, there are five ways of measuring the performance of a classifier:

- **Accuracy**: Confidence of the classifier, usually estimated as the percentage of test examples correctly classified over the total.

- **Efficiency**: Time consumed by the model when classifying a test example.

- **Interpretability**: Clarity and credibility, from the human point of view, of the classification model. The higher the interpretability of the produced model is, the more knowledge can be extracted from it.

- **Learning speed**: Time required by the machine learning algorithm to build the clasiffication model.

- **Robustness**: Minimum number of examples needed to obtain a precise and reliable classification model.

There are many different approaches proposed in the literature to perform classification tasks, including statistical techniques, discriminant functions, neural networks, decision trees, support vector machines and many more. Among them, there is a subclass of algorithms which can be highlighted by its unique ability of performing the learning process using directly the examples provided in the training set: the instance based learning algorithms [AKA91].

The most well-known instance based algorithm is the nearest neighbor classifier. Its is a non-parametric method for pattern classification [DHS00, HTF09]. Introduced by Fix and Hodges in 1951 [FH51], the nearest neighbor classifier gained considerable popularity after 1967, when some of its formal properties were described by Cover and Hart [CH67]. Cover's work was the key milestone of a subject which now has become a lively research field for many researchers in pattern recognition and machine learning areas: the study and development of one of the top ten algorithms in data mining [WK09].

The nearest neighbor classifier is also an important example of *lazy learning* algorithms [Aha97]. Lazy learning algorithms are techniques which, in contrast with the rest of machine learning approaches, do not built a model in its training phase. They can be defined by three main properties:

- They defer the processing of the training data until the test phase. Thus, usually there is no training phase when pure lazy learning algorithms are used.
- Its output is obtained as a combination of their stored data. Hence, regardless the absence of a model, the training data is still a key element in the performance of lazy learning algorithms.
- The information obtained when processing the output is discarded. Pure lazy learning algorithms do not learn anything from previous classification steps.

As a lazy learning algorithm, the nearest neighbor classifier has inherited many beneficial traits, including its relative simplicity, its adaptability to many different general problems, its flexibility to be optimized, for example, by modifying its similarity function, the lack of the necessity of a training phase and the capability of incorporating new knowledge to the classifier easily (this step, immediate for nearest neighbor classifiers, usually implies to rebuild a new model for most machine learning approaches). However, the nearest neighbor classifier also suffers from several problems:

- In test phase is computationally costly, since it has to analyze the whole training set every time a new instance is classified.
- In its original definition, it is not able to process nominal data. Also, it cannot handle missing values (incomplete information in the instances of the training data).
- It lacks of interpretability capabilities, since there is no model to explain how the classification is performed.
- Is not tolerant to noise. Noisy (incorrect) examples may affect the behavior of the classifier if there are many in the surrounding of the test instance.
- It is very sensitive with respect to the similarity measure chosen to discriminate between the instances.
- It is affected by the presence of irrelevant attributes describing the data.

- With most of the common similarity measures, it is affected by the so-called *curse of dimensionality*, which means that the nearest neighbor classifier does not measure similarities in a proper way when the dimensionality of the data growths (that is, when the number of attributes describing the training data is too large).

Many approaches have been developed in the last decades in order to tackle these issues. A remarkable number of them are based on Soft Computing approaches, incorporating the use of evolutionary computation, approximate reasoning, fuzzy sets and so on to the enhancement of the nearest neighbor classifier.

The present thesis is developed following this path. Advanced models for nearest neighbor classification will be designed, based on Soft Computing techniques. A remarkable number of them will be based on data preprocessing and data reduction techniques, both working over the training data instead on just modifying the nearest neighbor classifier. To define them, evolutionary algorithms, fuzzy rough sets and fuzzy sets will be incorporated as useful tools for improving the performance of the classifier, in the sense of increasing the accuracy of the classification and also in the sense of improving the efficiency of the test classification phase.

After this introduction, the next section (Section 2.) is devoted to describe in detail three main areas related: Data preprocessing and data reduction for nearest neighbor classification (Section 2.1), evolutionary algorithms and coevolution (Section 2.2) and fuzzy sets and fuzzy rough sets (Section 2.3). All of them are fundamental areas for defining and describing the proposals presented as a results of this thesis.

Next, the justification of this memory will be given in Section 3., describing the open problems addressed. The objectives pursued when tackling them are described in Section 4.. Section 5. presents a summary on the works that compose this memory. A joint discussion of results is provided in Section 6., showing the connection between each of the objectives and how have been reached each of them. A summary of the conclusions drawn is provided in Section 7.. Finally, in Section 8. we point out several open future lines of work derived from the results achieved.

The second part of the memory is constituted by seven publications, organized into four sections (related with the four sections in which the results and conclusions are reported). These publications are the following:

- Evolutionary Algorithms in Prototype Reduction:
  - A Survey on Evolutionary Instance Selection and Generation.
- Coevolutionary Algorithms for Enhancing Nearest Neighbor Classification:
  - IFS-CoCo: Instance and Feature Selection Based on Cooperative Coevolution With Nearest Neighbor Rule.
  - Integrating Instance Selection, Instance Weighting, and Feature Weighting for Nearest Neighbor Classifiers by Coevolutionary Algorithms.
- Fuzzy Rough Sets and Evolutionary Algorithms in Nearest Neighbor Classification:
  - Enhancing Evolutionary Instance Selection Algorithms by Means of Fuzzy Rough Set Based Feature Selection.
  - On the Use of Evolutionary Feature Selection for Improving Fuzzy Rough Set Based Prototype Selection.

- Fuzzy Nearest Neighbor Classification:
  - Fuzzy Nearest Neighbor Algorithms: Taxonomy, Experimental analysis and Prospects.
  - EIVF-kNN: An Evolutionary Interval Valued Fuzzy K-Nearest Neighbors Classifier.

Introducción

En los últimos años, se ha producido un gran incremento en la cantidad de datos que se deben procesar en aplicaciones en industria, medicina, economa y muchas otras áreas. Los investigadores y profesionales de muchas áreas están encontrando una necesidad creciente de nuevos desarrollos para recolectar, analizar y entender la información que se almacena continuamente en grandes bases de datos. La gestión y el análisis de esta cantidad de datos está por encima de las capacidades humanas y, por tanto, es necesario confiar en procedimientos automáticos para adquirir conocimientos valiosos, normalmente ocultos en esta *avalancha de datos* contemporánea. [BHS09].

Por ello, el análisis de datos y la extracción de conocimiento útil se ha convertido en una tarea obligada - y en un difcil desafo - para muchos procesos de investigación y de negocio en la actualidad. Este proceso, conocido formalmente como *Descubrimiento de Conocimiento en Bases de Datos* (DCBD) [FPSS96, Han05] comprende la tarea de obtener nuevo conocimiento a partir de grandes cantidades de datos. Estos nuevos fragmentos de conocimiento deben ser útiles y válidos, y deben tener sentido para el problema especfico abordado.

El proceso de DCDB se define generalmente como una sucesión de cinco etapas:

- Selección de los datos sobre los que se aplicará el proceso de DCDB.
- Preprocesamiento de los datos, incluyendo varias fases de limpieza y reducción para facilitar el desarrollo de las etapas subsecuentes.
- Transformación de los datos en una representación estructurada; una representación del conocimiento del problema almacenado en los datos iniciales tan precisa como sea posible.
- Construcción de nuevos patrones de conocimiento, a partir del análisis de la representación construida en el paso anterior.
- Visualización de los datos, describiendo y comprendiendo los patrones obtenidos, de forma que puedan ser útiles para los usuarios.

Siguiendo estas etapas, el proceso de DCDB incluye un gran número de técnicas y metodologas de las ciencias de la computación y la inteligencia artificial, para abordar numerosas tareas en cada paso. Sus etapas centrales también son definidas mediante el término *minera de datos* [TSK05, WFH11]. La minera de datos es la disciplina centrada en la identificación de patrones y la predicción de relaciones entre los datos. Normalmente, sus técnicas se pueden clasificar como descriptivas (cuando son aplicadas a descubrir patrones interesantes entre los datos) o predictivas (cuando se predice el comportamiento de un modelo a través del análisis de los datos disponibles).

Los procesos descriptivos y predictivos de la minera de datos son conducidos mediante algoritmos de *aprendizaje automático* [MD01, Alp10, WFH11], introducidos como mecanismos capaces de inducir conocimiento a partir de datos. Dicha inducción de conocimiento es deseable en problemas que no dispongan de una solución algortmica eficiente, definidos vagamente, o especificados de forma informal.

El uso de algoritmos de aprendizaje automático presenta dos vertientes: Pueden ser empleados simplemente como cajas negras, obteniéndose como resultado tan solo las salidas de los modelos. Sin embargo, algunos algoritmos pueden ser empleados como herramientas de representación de conocimiento, construyendo una estructura simbólica de conocimiento dispuesta a ser útil desde al punto de vista de la funcionalidad, pero también desde la perspectiva de la interpretabilidad.

Dependiendo de sus objetivos, los algoritmos de aprendizaje automático pueden ser clasificados en dos áreas diferentes:

- **Aprendizaje supervisado**: Son algoritmos centrados en emplear un conjunto de ejemplos etiquetados, describiendo información de varias variables de entrada, para predecir el valor de varias variables de salida. La categoras más comunes de algoritmos de aprendizaje supervisado son clasificación (donde la variable a predecir es discreta; por ejemplo rojo, verde, azul) y regresión (donde la variable a predecir es continua; por ejemplo: temperatura, peso ...
- **Aprendizaje no supervisado**: Son algoritmos centrados en buscar nuevos patrones y relaciones en datos no etiquetados. La categoras más comunes de algoritmos de aprendizaje no supervisado son agrupamiento (el proceso de separar datos en varios grupos, manteniendo ejemplos de los mismos grupos tan similares entre s como sea posible) y asociación (la identificación de relaciones en datos transaccionales).

Los métodos de clasificación se pueden definir como técnicas que permiten aprender como categorizar elementos dentro de varias clases predefinidas. Un clasificador recibe un conjunto de datos como entrada, definido como *conjunto de entrenamiento*, y aprende un modelo de clasificación con él. En el proceso de validación del clasificador, se emplea un conjunto adicional de ejemplos, no contemplados durante el proceso de aprendizaje (el *conjunto de test*) para comprobar la precisión del clasificador.

Generalmente, existen cinco maneras de medir el rendimiento de un clasificador:

- **Precisión**: Confianza del clasificador, normalmente estimada como el porcentaje de ejemplos de test correctamente clasificados sobre el total.
- **Eficiencia**: Tiempo consumido por el clasificador a la hora de clasificar un ejemplo de test.
- **Interpretabilidad**: Claridad y credibilidad, desde el punto de vista humano, del modelo de clasificación. Cuanto más alta sea la interpretabilidad del modelo producido, mayor cantidad de conocimiento podrá ser extrada de él.
- **Velocidad de aprendizaje**: Tiempo requerido por el algoritmo de aprendizaje automático para construir el modelo de clasificación.
- **Robustez**: Número mnimo de ejemplos necesarios para obtener un modelo de clasificación preciso y fiable.

En la literatura existen muchas propuestas diferentes para la realización de tareas de clasificación, incluyendo técnicas estadsticas, funciones discriminantes, redes neuronales, árboles de decisión, máquinas de vectores soporte y muchas otras. Entre ellas, existe una subclase de algoritmos que merece ser destacada por su habilidad única de realizar el proceso de aprendizaje directamente a partir de los ejemplos incluidos en el conjunto de entrenamiento: los algoritmos de aprendizaje basados en instancias [AKA91].

El clasificador del vecino más cercano es el algoritmo basado en instancias más conocido. Es un método no paramétrico de clasificación de patrones [DHS00, HTF09]. Introducido por Fix y Hodges en 1951 [FH51], el clasificador del vecino más cercano adquirió una popularidad considerable tras 1967, cuando varias de sus propiedades formales fueron descritas por Cover y Hart [CH67]. El trabajo de Cover fue la piedra angular de una materia que ahora se ha convertido en un vivo campo de investigación para muchos investigadores de las áreas de reconocimiento de patrones y aprendizaje automático: el estudio y desarrollo de uno de los diez algoritmos más importantes en minera de datos [WK09].

El clasificador del vecino más cercano es también un importante ejemplo de algoritmos de *aprendizaje perezoso*. Los algoritmos de aprendizaje perezoso son técnicas que, a diferencia del resto de técnicas de aprendizaje automático, no construyen un clasificador durante su fase de entrenamiento. Pueden ser definidos mediante tres propiedades:

- Retrasan el procesamiento de los datos de entrenamiento hasta la fase de test. Por tanto, en los modelos de aprendizaje perezoso puro no hay fase de entrenamiento.
- Su salida es obtenida como una combinación de los datos obtenidos. Por eso, a pesar de ausencia de un modelo, los datos de entrenamiento siguen siendo un elemento clave en el rendimiento de los algoritmos de aprendizaje perezoso.
- La información obtenida al procesar la salida se descarta. Los algoritmos de aprendizaje perezoso puro no aprenden nada de pasos previos del proceso de clasificación.

Como algoritmo de aprendizaje perezoso, el clasificador del vecino más cercano ha heredado muchos rasgos beneficiosos, incluyendo su relativa simplicidad, su adaptabilidad a diferentes problemas generales, su flexibilidad a la hora de ser optimizado, por ejemplo, mediante la optimización de su función de similaridad, la no necesidad de un proceso de entrenamiento y la capacidad de poder incorporar nuevo conocimiento al clasificador de forma sencilla (este paso, inmediato en el caso del clasificador del vecino más cercano, normalmente implica la reconstrucción de un nuevo modelo para la mayora de métodos de aprendizaje automático). Pese a todo, el clasificador del vecino más cercano también sufre debido a varios problemas:

- Es computacionalmente costoso en la fase de test, dado que debe analizar el conjunto de entrenamiento completo cada vez que una nueva instancia es clasificada.
- En su definición original, no es capaz de procesar datos nominales. Además, tampoco puede manejar valores perdidos (información incompleta en los datos de entrenamiento).
- No dispone de capacidades de interpretabilidad, dado que no hay un modelo para explicar cómo se ha realizado la clasificación.
- No es tolerante al ruido. Los ejemplos ruidosos (incorrectos) pueden afectar al comportamiento del clasificador si hay muchas instancias incorrectas en las proximidades de la instancia de test.
- Es muy sensible con respecto a la medida de similaridad escogida para discriminar entre las instancias.

- Se ve afectado por la presencia de atributos irrelevantes describiendo los datos.
- Con la mayora de las medidas de similaridad comunes, se ve afectado por la, as llamada, *maldición de la dimensionalidad*, lo que significa que el clasificador del vecino más cercano no mide de forma correcta la similaridad entre las instancias cuando la dimensionalidad de los datos crece demasiado (es decir, cuando el número de atributos describiendo los datos es demasiado grande).

En las últimas décadas se han desarrollado muchas propuestas para abordar estos problemas. Un número destacable de ellas están basadas en técnicas de Computación Flexible, incorporando el uso de la computación evolutiva, el razonamiento aproximando, los conjuntos difusos y otras técnicas en la mejora del clasificador del vecino más cercano.

La presente tesis de desarrolla siguiendo este camino. Se diseñarán modelos avanzados para clasificación del vecino más cercano, basado en técnicas de Computación Flexible. Un número destacable de ellas están basadas en técnicas de preprocesamiento y reducción de datos, trabajando sobre los datos de entrenamiento en lugar de simplemente modificar el clasificador del vecino más cercano. Algoritmos evolutivos, conjuntos rugosos difusos y conjuntos difusos seran incorporados como herramientas útiles a la hora de definir estas técnicas para la mejora del rendimiento del clasificador, en el sentido de incrementar la precisión de la clasificación y de mejorar su eficiencia en la fase de test.

Tras esta introducción, la siguiente sección (Sección 2.) está dedicada a describir en detalle tres principales áreas relacionadas: Preprocesamiento y reducción de datos para clasificación del vecino más cercano (Sección 2.1), algoritmos evolutivos y coevolución (Sección 2.2) y conjuntos difusos y conjuntos rugosos difusos (Sección 2.3). Todas ellas son áreas fundamentales para definir y describir las propuestas presentadas como resultado de esta tesis.

Después, la justificación de esta memoria se presenta en la Sección 3., describiendo los problemas abiertos abordados. Los objetivos perseguidos al abordar dichos problemas son descritos en la Sección 4.. La Sección 5. presenta un resumen de los trabajos que componen esta memoria. Se aporta una discusión conjunta de resultados en la Sección 6., mostrando la conexión entre cada uno de los objetivos y como ha sido alcanzado cada uno de ellos. En la Sección 7. se incluye un resumen de las conclusiones alcanzadas. Finalmente, en la Sección 8. se destacan varias lneas de trabajo futuro abiertas, derivadas de los resultados alcanzados.

La segunda parte de la memoria se constituye de siete publicaciones, organizadas en cuatro secciones (relacionadas con las cuatros secciones en que los resultados y conclusiones son mostrados). Estas publicaciones son las siguientes:

- Algoritmos evolutivos en reducción de prototipos:
  - A Survey on Evolutionary Instance Selection and Generation.
- Algoritmos coevolutivos para la mejora de la clasificación del vecino más cercano:
  - IFS-CoCo: Instance and Feature Selection Based on Cooperative Coevolution With Nearest Neighbor Rule.
  - Integrating Instance Selection, Instance Weighting, and Feature Weighting for Nearest Neighbor Classifiers by Coevolutionary Algorithms.
- Conjuntos rugosos difusos y algoritmos evolutivos en la clasificación del vecino más cercano:

  - Enhancing Evolutionary Instance Selection Algorithms by Means of Fuzzy Rough Set Based Feature Selection.
  - On the Use of Evolutionary Feature Selection for Improving Fuzzy Rough Set Based Prototype Selection.

- Clasificación del vecino más cercano difusa:
  - Fuzzy Nearest Neighbor Algorithms: Taxonomy, Experimental analysis and Prospects.
  - EIVF-kNN: An Evolutionary Interval Valued Fuzzy K-Nearest Neighbors Classifier.

2. Preliminaries

Three major fields in machine learning and Soft Computing are presented in this section, as preliminary background on the topics in which the thesis are elaborated. These fields are *data preprocessing and data reduction for nearest neighbor classification* (Section 2.1), *evolutionary algorithms and coevolution* (Section 2.2) and *fuzzy sets and fuzzy rough sets* (Section 2.3)

2.1 Data preprocessing for nearest neighbor classification

The preparation of the data that composes the training set [Pyl99] is a capital issue in the performance of nearest neighbor classifiers. Obviously, the more representative the training data would be, the more accurate the nearest neighbor rule will perform. However, the analysis on the quality of the training data does not have to be limited to the process of its acquisition.

Data preprocessing techniques play a crucial role in the preparation, adjustment and enhancement of the performance of nearest neighbor classifiers. Working on the initial data available, these techniques can be applied to tackle several common problems, such as the presence of harmful data.

Regardless of its representativity, real-world data poses a challenge for machine learning algorithms, including nearest neighbor classifiers. Noisy data, duplicate or inconsistent instances, outliers, missing or incomplete instances, and simply erroneous data are some examples of the difficulties that may be overcome by introducing data preparation techniques. Moreover, they can be used to reduce the size of the training set by discarding irrelevant data, which will help to improve the performance of the classifier with respect to its efficiency.

Data reduction is one of the most successful ways of performing data preparation for nearest neighbor classifiers. This field encloses several different techniques, being the most remarkable the following:

- **Instance selection**: The selection of the most representative instances in training data [LM01]. Instance selection has been the focus of many research approaches in the last decades, given its capabilities of improving the performance of classification algorithms thorough the simultaneous reduction of the training set size and the removal of noisy and irrelevant examples. Instance selection approaches can be classified as training set selection techniques (instance selection for training sets used in classifiers' models construction) or prototype selection techniques (instance selection for selecting prototypes for nearest neighbor classifiers).
- **Feature selection**: The selection of the most representative attributes in the domain of the problem [LM07]. Feature selection is a very interesting data reduction technique, since

it allows the classification model to focus on an specific subset of features, increasing its generality and also its efficiency due to the reduction of the number of attributes per instance to process. Furthermore, it also helps to deal with the *curse of dimensionality* in nearest neighbor classification.

- **Instance generation**: Instance generation is an extension of instance selection which in addition to selecting data can be applied to generate new artificial examples [TDGH12]. These new examples can be very useful in the task of filling regions of the domain of the problem with no representative examples in original data. Besides the creation of new examples, instance generation focus is also on cleaning the initial training data, reducing substantially its original size. Most of the existing approaches for instance generation are specifically developed for improving nearest neighbor classifiers, composing the field of prototype generation.
- **Feature generation**: Also known as feature extraction [GGNZ06], this technique offers several approaches for generating new attributes to describe the training data (which may be added or not to some of the original ones). Here, the focus is to obtain a final subset of features able to represent the data in a better suited way for the classification process. Linear and non-linear transformations or statistical techniques such as principal component analysis [Jol02] are classical examples of these techniques.
- **Feature discretization**: Discretization [GLS^{+}12] is mainly considered as a preprocessing tool for transforming quantitative data into qualitative data, enabling symbolic data mining algorithms (for example, naive bayes [DP97]) to be applied over continuous data. However, it can also be considered as a data reduction technique, given its capabilities for modifying the representation of information, making it more concise and specific.

Among these techniques, prototype selection [GDCH12] and feature selection [LY05] can be highlighted as the most useful ones for improving nearest neighbor classification. Particularly, prototype selection is the most considered in the literature, featuring many different approaches for the reduction and cleaning of the reference data.

Depending on the strategy followed, prototype selection methods can be categorized into three classes: preservation methods, which aim to obtain a consistent subset from the training data, ignoring the presence of noise; noise removal methods, which aim to remove noise both in the boundary points (instances near to the decision boundaries) and in the inner points (instances far from the decision boundaries), and hybrid methods, which perform both objectives simultaneously. Although their performance may be different depending on the specific problem tackled, hybrid methods generally shows an outstanding behavior when improving nearest neighbor classifier.

On the other hand, feature selection approaches can also be categorized by several properties. The most usual, and the one by which categorize them here is depending on the mechanism followed to assess the quality of a given subset of features. This enable us to define three categories of feature selection methods:

- **Filter methods**, where the selection criterion is independent of the classifier. Instead, feature subsets' quality is evaluated through the use of descriptive functions, such as separability measures [GE03].
- **Wrapper methods**, where the selection criterion is dependent on the classifier. The performance of the classifier itself is considered to evaluate the quality of every subset of features considered by the feature selection algorithm [KJ97].

- **Embedded methods**, where the search for an optimal subset of features is built into the classifier construction [SIL07].

Among these options, both wrapper and filter approaches are suitable for nearest neighbor classifiers, although wrapper methods usually offer a better performance, at the cost of an increase in the computational cost of the feature selection search procedure.

However, data reduction approaches are not the only preprocessing techniques of interest in nearest neighbor classification. Weighting schemes, introduced as a way of refining the selection schemes for features and instances, are often a very useful addition to nearest neighbor based classifiers.

Similarly to selection techniques, weighting schemes can be applied both to instances and features. Instance weighting schemes are usually introduced for modifying the importance of specific training instances in the computation of the distance measure. Sometimes this is performed to the whole data set, whereas other approaches apply the weights only when a test instance has been presented to the classifier (performing an *ad hoc* local modification directly focused on the specific test instance) [AMS97].

On the other hand, feature weighting schemes are also very popular in the task of improving nearest neighbor classification [WAM97]. In this class of weighting algorithms, weights are assigned to features in an attempt to relax the conditions of feature selection. Instead of just selecting or removing a feature, feature weighting schemes allows to estimate a degree of importance of each feature with respect to the classification process. This enables the classifier to consider information about most of the features of the problem, but giving more importance to the most relevant ones.

2.2 Evolutionary algorithms and coevolution

Evolutionary algorithms are techniques inspired by natural computation, designed for tackling problems of search and optimization [ES03, GJ05, PF09]. In the literature it is possible to find many different evolutionary techniques, working all of them by representing a set of solutions, which is evolved as the algorithm is carried out. During the evolution, different evolutionary operators are applied to modify, combine and distribute the solutions, until the evolutionary process is stopped.

The introduction of evolutionary algorithms has been very beneficial in the task of enhancing data mining algorithms [Fre02]. The good adaptability of evolutionary algorithms, which are able to tackle both binary and real valued problems as long as they could be formulated as a search procedure, is responsible of their use on upgrading and adjusting many different machine learning algorithms. As an example, the task of adapting the parameters of the construction of a machine learning model to a given problem can be formulated as a search process, and therefore it can be performed by an evolutionary algorithm.

Of course, this also includes nearest neighbor classification. However, there is another use of evolutionary algorithms for improving the nearest neighbor classifier, which is the application of evolutionary data reduction techniques.

Among them, evolutionary instance selection is probably the field that deserves more importance in nearest neighbor classification. The first works in this line were presented by Kuncheva et al. in 1995 and 1998 [Kun95, KB98]. They featured evolutionary prototype selection approaches, using binary codification for representing the selection of prototypes of the training set (where 1 is used to denote that a prototype is selected, whereas 0 denotes that it is not chosen).

The field of evolutionary prototype selection was growing until 2003, when Cano et al. presented a work [CHL03] in which the way in which the search of prototypes was updated. Their search algorithms included the consideration of the reduction rate (the rate of prototypes not selected from the training set) in the evaluations of solutions.

By incorporating a weighted average of the accuracy of the classifier (using a wrapper model) and the reduction rate, the approaches presented in [CHL03] were able to balance both objectives along the search, achieving an effective trade-off between accuracy in test classification and efficiency of the classifier. Due to its significance, this approach has been adopted by many evolutionary approaches for prototype selection until nowadays [GCH08, DGH10].

Another interesting branch of evolutionary computation is focused on coevolutionary algorithms. Coevolution is the field of evolutionary computation which deals about algorithms able to manage two or more populations simultaneously. These population coexist, interacting between them throughout all the evolutionary process.

Traditionally, coevolution has been used as a way of splitting the representation of a problem into different parts, using a population to handle each one separately. This allows the algorithm to follow a divide-and-conquer strategy, where each population can focus its efforts on a part of the problem. If the solutions obtained by each population are properly joined, and the interaction between individuals is managed in a suitable way, the coevolutionary model can show an outstanding performance in its application.

In coevolution, the interaction between individuals of different populations is key to the success of the search. Coevolution is often divided into two classes, regarding the type of interaction defined:

- **Cooperative coevolution**: Where each population evolves individuals representing a component of the final solution. Full candidate solutions are obtained by joining an individual chosen from each population. In this way, increases in a collaborative fitness value are shared among individuals of all populations [PJ00].
- **Competitive coevolution**: Where 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 [RB97].

Coevolution has given birth to several approaches to the construction of nearest neighbor classifiers (for example [GP07]). Some of them also considers the inclusion of data reduction techniques, such as [GPdCOB10], where evolutionary instance selection is carried out by means of a cooperative coevolutionary algorithm. This is an example of how cooperative coevolution and data reduction approaches can be applied to enhance the performance of nearest neighbor classifier, although it would be possible to improve further the classifier, if coevolution is used to integrate several data reduction approaches into the algorithm, instead of just sticking to instance selection.

2.3 Fuzzy sets and fuzzy rough sets

Fuzzy sets are sets whose elements have degrees of membership. They were introduced originally by Zadeh [Zad65] in 1965, as an extension of the classical mathematical notion of set. In classical set theory, the membership of elements in a set is defined in binary terms: an element either belongs or does not belong to the set. By contrast, fuzzy set theory allows the definition of gradual membership of elements in a set. These gradual memberships can be described with the aid of a membership function valued in the real unit interval [0, 1].

Fuzzy set theory can be incorporated into a wide range of domains in which information is incomplete or imprecise. This includes classification problems, where many sources of uncertainty can be identified: The decision rule that chooses the class to which every test instance belongs, the inner mechanisms of many machine learning models, and even the definition of the classes of the instances of the training set are potential sources of uncertainty.

Using the capability of fuzzy sets for modeling uncertainty, several approaches to the management of uncertainty in nearest neighbor classification have been presented. Most of the early efforts [Jow83, KGG85] have been directed to characterize the uncertainty on the memberships of the instances to the classes. In contrast with the classical definition of the training set in classification, where an instance belongs only to a single class, fuzzy sets theory allows to model the membership of instances to several classes simultaneously, providing with more information about the status of the instance to the classifier.

Lately, extensions and advanced versions of fuzzy sets have also been incorporated to address the problem of managing uncertainty in many of the stages of the nearest neighbor classifier. The most relevant examples includes possibilistic theory [Zad78], intuitionistic fuzzy sets [Ata86] and type-2 fuzzy sets [Men07], all of them focuses on improving the capabilities of the classic nearest neighbor rule.

On the other hand, rough set theory [Paw82] has been also introduced to tackle several data mining problems. Generally speaking, rough sets provide with a formalism to define indiscernibility relations between the information represented in a decision system; this is equivalent to highlight inconsistencies in the training data of a classification problem with respect to the attributes that define each instance and its class assigned.

Recently, fuzzy rough sets [CCK07] have been also introduced, in an effort to increase the generality of classic rough sets based approaches. One of the drawbacks of the classic approaches is that they must work over discrete data. This means that if it is necessary to apply such techniques over continuous data, a discretization process [GLS$^+$12] must be applied at a previous step.

Although discretization can be a good technique for solving this kind of issues, for nearest neighbor classifiers the use of continuous values is still preferred due to the intrinsic characteristics of most of the similarity measures used. Hence, fuzzy rough sets, the hybridization between rough sets and fuzzy sets, are a better choice in this scenario.

Very recently, data reduction approaches based on fuzzy rough sets have been presented. In [CJHS10] is presented a method for performing feature selection. This method, based on the concept of discernibility, analyzes the instances of a training set, identifying which are discernible (with respect to the elements belonging to the other classes of the domain), regarding the specific set of features considered. In this way, a measure of the quality of a subset of features is defined, enabling to select those features which are able to fully discern all the instances of the training set (or, at least, discern them as much as possible). In this way, the pruned training set can maintain its capabilities of separating instances belonging to different classes (or even increase them, by the removal of noisy and irrelevant features), while its size is reduced.

Fuzzy rough sets based instance selection approaches can follow a similar approach. In [JC09], the idea of the fuzzy-rough positive region is formulated, considering that instances on the border of a class (that is, for which there exists a similar instance in another class) will have a small membership value to the fuzzy-rough positive region compared to instances in the center of a class.

This is an orthogonal way of using fuzzy rough sets for measuring the quality of an instance as a typical representative of its class. Thus, instances with a low quality can be determined as noise, and can be discarded by the instance selection process.

Both theories, fuzzy sets and rough sets, can be used to improve the way in which nearest neighbor classifiers, through the redefinition of some of their classical mechanisms or through the use of data reduction. Both are examples of strategies worth to be explored in the definition of advanced models for nearest neighbor classification, able to improve the behavior of the classic nearest neighbor rule.

3. Justification

As it has been shown in the first section, Soft Computing techniques have arisen as useful tools for improving the way in which the nearest neighbor rule behaves. Advanced evolutionary techniques - such as cooperative coevolution - fuzzy rough sets and fuzzy sets have the potential of becoming a helpful enhancement in the task of adjusting and improving the performance of nearest neighbor classifiers.

If these techniques are to be adopted for developing data based approaches, the following key issues should be addressed:

- Very recently, there is a important number of evolutionary proposals for instance selection and generation. They should be analyzed in detail, given its flexibility and good performance in both problems. Specifically, in the field of prototype selection, evolutionary approaches are highlighted as the best branch of techniques of the current state of the art [GDCH12].

- However, most of the current evolutionary approaches only focus in performing a single data preprocessing task. It would be very desirable to develop advanced evolutionary techniques, able to develop several data preprocessing tasks at once, making the most of the existing synergy between them.

  This issue can be addressed with coevolutionary algorithms. Although the traditional way in which coevolution and cooperative coevolution are presented is as tools for performing domain decomposition (even when they are applied to instance selection problems [GPdCOB10]), it turns out that cooperative coevolution can also be used to performing several data preprocessing tasks simultaneously, providing that their solutions could be evaluated in an homogeneous way.

- Another interesting trend would be to combine evolutionary algorithms with other outstanding Soft Computing approaches for data preprocessing. In this point, fuzzy rough sets arise as a successful technique, mostly in the field of feature selection. Their flexibility and low computational cost (compared with evolutionary techniques, in general) makes them desirable for the task of developing hybrid data preprocessing approaches, combining both fuzzy rough sets and evolutionary algorithms.

- Finally, outside of the data preprocessing area there are plenty of approaches for the enhancement of the nearest neighbor rule. Among them, fuzzy nearest neighbor algorithms provide with mechanisms for managing uncertainty within the classification process. Perhaps the most useful trait of these approaches is the capability of defining the class of the prototypes in the training data as membership functions, which allow to manage borderline, dubious and noise examples as members of several classes at once (with varying degrees of confidence). Hence, a throughout study of this family of algorithms is essential in order to be able to develop robust classifiers based on the nearest neighbor rule.

All this issues refers to a common topic, which is also the main theme of this thesis: The development of advanced models for nearest neighbor classification with the support of Soft Computing techniques.

4. Objectives

After studying the current state of all the areas described in the previous sections, it is possible to focus on the actual objectives of the thesis. They will include the research and analysis of the background fields described before, and the development of advanced models for nearest neighbor classification based on their most promising properties of each field. More specifically, the objectives are:

- **To study the current state of the art in evolutionary instance selection**. This would mainly include evolutionary prototype selection, but evolutionary prototype generation and evolutionary training set selection will be also covered, in order to have a full understanding on the capabilities of evolutionary instance selection as a whole.

- **To analyze cooperative coevolution and develop new cooperative coevolution data preprocessing models**. These models should make use of the best capabilities of cooperative coevolution, incorporating several data preprocessing tasks into a single algorithm. Cooperation between the different tasks should be the key element in the new models, enabling to combine different strategies to obtain very accurate solutions at the end of the evolutionary process.

- **To incorporate fuzzy rough sets based preprocessing techniques**. After studying the last works published using fuzzy rough sets for data reduction, it would be interesting incorporate them to a general evolutionary framework. Two possibilities are opened. The first one deals with performing feature selection using fuzzy rough sets techniques, within the framework of an evolutionary prototype selection model, whereas the second one would be to develop the orthogonal model - a fuzzy rough sets based prototype selection algorithm embedded into a feature selection evolutionary framework. A full study of both approaches would help to characterize the usefulness of the combination of both evolutionary algorithms and fuzzy rough sets, when improving nearest neighbor classification through data reduction.

- **To review and contribute to the state of the art in fuzzy nearest neighbor classification**. Given the possibilities opened with fuzzy nearest neighbor algorithms and their unique capabilities, a last step in the development of Soft Computing based nearest neighbor classifiers would be to analyze extensively the field of fuzzy nearest neighbor classification. After a throughout analysis, the most promising characteristics of these techniques could be chosen to develop a new technique, able to both show the possibilities of evolutionary algorithms and fuzzy sets for building an advanced nearest neighbor classifier and to contribute to the state of the art in fuzzy nearest neighbor classification.

5. Summary

This thesis is composed by seven works, organized into four different parts. Each part is devoted to pursue one of the objectives described, contributing as a whole in the development of several advanced nearest neighbor classifiers based on Soft Computing techniques.

- Evolutionary Algorithms in Prototype Reduction.
- Coevolutionary Algorithms for Enhancing Nearest Neighbor Classification
- Fuzzy Rough Sets and Evolutionary Algorithms in Nearest Neighbor Classification
- Fuzzy Nearest Neighbor Classification

This section shows a summary of the different proposals presented in this dissertation, describing the associated publications and their main contents.

5.1 Evolutionary Algorithms in Prototype Reduction

The use of evolutionary algorithms to perform data reduction tasks has become an effective approach for improving the performance of machine learning techniques, including nearest neighbor classifiers. Many proposals in the literature have shown that evolutionary algorithms obtain excellent results in their application as instance selection and instance generation procedures.

A review has been performed, highlighting the features of evolutionary instance selection algorithms (including evolutionary prototype selection and evolutionary training set selection approaches) and evolutionary instance generation algorithms.

The field of evolutionary prototype selection has been analyzed thoroughly, starting from the very first proposals [Kun95, KB98] and analyzing the key milestone in the area, the work of Cano et al. [CHL03]. Common characteristics of these and other proposals have also been reviewed, including SSMA [GCH08], an evolutionary prototype selection algorithm which later would become the best method in the state of the art in prototype selection [GDCH12].

The review also analyzed several hybrid methods, such as IGA [HLL02] and HGA [RGPC08] (two methods that perform prototype and feature selection encoding both subsets in the solutions of genetic algorithms) or GoCBR [AK09] (a hybrid method for prototype selection and feature weighting).

Regarding training set selection, the review has covered several approaches for the enhancement of decision trees (for example, [CHL07]), neural networks [INN01, Kim06] and to optimize the results of subgroup discovery processes [CHLG08, CGH08].

With respect to evolutionary instance generation, various approaches have been analyzed including those using different evolutionary techniques such as particle swarm optimization [KES01] (PSO [NL09] and AMPSO [CGI09] prototype generation algorithms) and artificial immune systems [Das98] (the PSCSA approach [Gar08]).

Finally, the review is concluded with the description of two challenging problems in machine learning, the scaling up problem [PK99, DGW02] and the imbalance data sets problem [BPM04, HG09] and the illustration of how evolutionary instance selection algorithms can be applied to tackle them, by splitting the data through stratification in the former problem [CHL05] and by performing evolutionary undersampling in the latter [GH09]

The journal article associated to this part is:

- J. Derrac, S. García, F. Herrera, A Survey on Evolutionary Instance Selection and Generation. International Journal of Applied Metaheuristic Computing 1:1, 60-92, doi: 10.4018/jamc.2010102604

5.2 Coevolutionary Algorithms for Enhancing Nearest Neighbor Classification

Cooperative coevolution has become a very useful tool in the development of advanced nearest classification models. The two approaches that will be reviewed here are managed by cooperative coevolutionary algorithms, helping to combine the efforts of several data preprocessing approaches at once.

The first proposal features a combination of instance selection, feature selection and dual selection (instance and feature selection) into a single cooperative coevolutionary algorithm. IFS-CoCo (*Instance and Feature Selection based on Cooperative Coevolution*) is composed of three populations. The individuals of each one define a different type of baseline nearest neighbor classifier, depending on each populations characteristics.

Each population is focused on performing a basic data reduction task: The first population performs an instance selection process, the second population performs a feature selection process, and the third population performs a dual selection process.

Figure 1 shows an scheme of the model. The CHC evolutionary algorithm is used to conduct the search in every population. Populations are updated sequentially, and the best chromosome of each one is kept for the joint evaluation process.

Solutions are, then, evaluated jointly. The fitness function follows the same rationale as in [CHL03], performing a weighted sum of training accuracy and reduction rate. However, the accuracy is evaluated jointly (as is shown in Figure 1), in contrast with reduction, which is still computed only for each single chromosome.

The second proposal, CIW-NN (*Coevolution of Instance selection and Weighting schemes for Nearest Neighbor classifiers*) introduces the use of weighting schemes for instances and features. As a coevolutionary algorithm, it also introduces the novelty of combining different evolutionary algorithms (CHC and an steady-state genetic algorithm, in this case) with different codifications (binary and real codifications).

CIW-NN is also composed of three populations. The first one performs an instance selection process guided by the CHC algorithm (following the same definition than in IFS-CoCo). The second and third populations perform a feature weighting process and an instance weighting process respectively. Both processes select the best possible weights to further increase the leave-one-out classification performance of the underlying nearest neighbor classifier. To do so, their search processes are guided by a steady-state genetic algorithm updated with a crossover operator with multiple descendants [SLVH09].

Figure 5.2 shows an scheme of the CIW-NN model, composed of a representation of the three populations of the model (Figure 2(a)) and an scheme of the fitness assignation scheme (Figure 2(b)). Cooperation between populations is also achieved in CIW-NN through the fitness function, but, in this time, the reduction rate is only considered for the instance selection population. On the contrary, accuracy is computed for the solutions of the three populations, following a combination of three chromosomes provided by each of the three populations of the model.

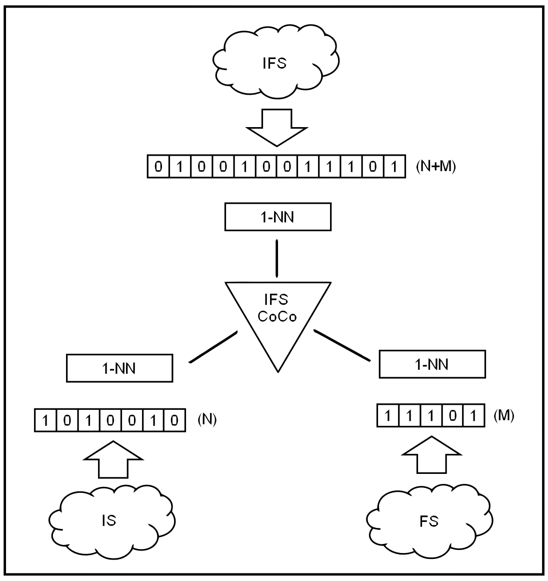


Figure 1: Population scheme of IFS-CoCo

Both models, IFS-CoCo and CIW-NN, show how cooperative coevolution can be used to build advanced nearest neighbor classifiers, improving the capabilities of the nearest neighbor rule through the use of both data reduction and weighting schemes.

The journal articles associated to this part are:

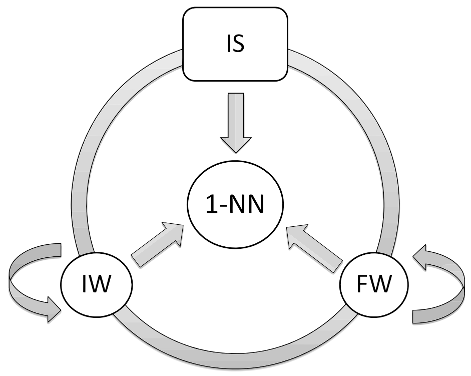
- J. Derrac, S. García, F. Herrera, IFS-CoCo: Instance and Feature Selection Based on Cooperative Coevolution With Nearest Neighbor Rule. Pattern Recognition, 43 (2010) 20822105 doi: 10.1016/j.patcog.2009.12.012

- J. Derrac, I. Triguero, S. García, F. Herrera, IFS-CoCo: Integrating Instance Selection, Instance Weighting, and Feature Weighting for Nearest Neighbor Classifiers by Coevolutionary Algorithms. IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics 42:5 (2012) 13831397 doi: 10.1109/TSMCB.2012.2191953

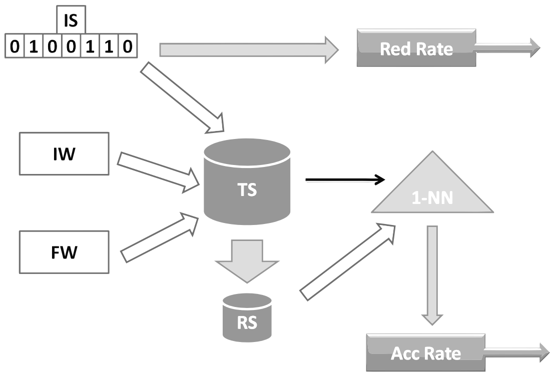
5.3 Fuzzy Rough Sets and Evolutionary Algorithms in Nearest Neighbor Classification

Fuzzy rough sets have been chosen as formalism for representing the extraction of minimal subsets of data, following a new approach for data reduction. Their use is preferred over classic rough sets given their greater flexibility, and their capability of being able to being applied directly over data sets representing continuous data without discretizing it at a previous step.

The approach presented in [CJHS10] provides with a proper tool for performing feature selection. It features a measure of to evaluate the discernibility of a subset of features, and an heuristic (the *Quickreduct* algorithm) to find the minimal reduct representing all the indiscernibility relations present in the training data. The features that form this reduct are chosen as the final subset of features obtained by the feature selection algorithm.



(a) Populations scheme



(b) Fitness computation

Figure 2: Description of the CIW-NN model

EIS-RFS (evolutionary instance selection enhanced by rough set based feature selection), the first hybrid approach developed, features an evolutionary instance selection procedure and a fuzzy rough sets based feature selection process. Instance selection is conducted by a steady-state genetic algorithm, which evolves at a somewhat slower pace than classical generational genetic algorithms. This property is very useful for combining its efforts with those of the feature selection procedure.

The feature selection, managed by *Quickreduct*, is performed every time a fixed number of evaluations has been spent by the evolutionary instance selection procedure. Its output is considered as the current subset of active features for the steady-state genetic algorithm. This means that the synergy between both techniques is achieved by modifying the domain (the features) considered by the evolutionary instance selection procedure at fixed points of the evolution. Consequently, the instances selected will depend heavily on the output of *Quickreduct*.

The model is completed with the inclusion of the VDM (value difference metric) distance [WM97] for nominal attributes, allowing EIS-RFS to tackle effectively both continuous and nominal data sets, regardless its origin.

On the other hand, fuzzy rough instance selection has been also developed recently [JC09]. These methods are based on the removal of instances that negatively affect the fuzzy positive region. Instances are removed until there is no uncertainty amongst them, that is, all remaining instances can be classified positively.

In our second approach, EFS-RPS (evolutionary feature selection for fuzzy rough set based prototype selection), fuzzy rough prototype selection and evolutionary feature selection are combined for developing an advanced data reduction model for nearest neighbor classification.

The quality of the instances in the training set is assessed following the concepts of fuzzy rough set theory. The membership of an instance to the positive region with respect to the current subset of features considered is used as a noise measure for it. Following this reasoning, a pruning procedure is carried out, iteratively removing prototypes until no improvement classification accuracy is found.

Evolutionary feature selection is performed by an steady-state genetic algorithm, considering both classification accuracy and reduction rate in the fitness function. However, only a minimal weight is given to the reduction rate in order to avoid excessive deletion of features to be favored by the by the fitness function.

The full EFS-RPS evolutionary model includes both reduction techniques: The fuzzy rough set based prototype selection procedure is performed, selecting which instances will be considered during the first stage of the evolutionary feature selection algorithm. Once several evaluations have been spent, the prototype selection algorithm is carried out again (considering the best subset of features found so far) to modify the environment of the feature selection procedure. This interweaving of prototype and feature selection phases is continued until the end of the algorithm.

With this definition, EFS-RPS can be defined as an orthogonal version of EIS-RFS, in which the roles of evolutionary algorithms and fuzzy rough set techniques is exchanged. This has enabled to develop two complementary models, each one with the potential of obtaining very promising results in the task of enhancing the performance of nearest neighbor classifiers.

The journal articles associated to this part are:

- J. Derrac, C. Cornelis, S. García, F. Herrera, Enhancing Evolutionary Instance Selection Algorithms by Means of Fuzzy Rough Set Based Feature Selection. Information Sciences, 186 (2012) 7392 doi: 10.1016/j.ins.2011.09.027
- J. Derrac, N. Verbiest, S. García, C. Cornelis, F. Herrera, On the Use of Evolutionary Feature Selection for Improving Fuzzy Rough Set Based Prototype Selection. Soft Computing 17:2 (2013) 223-238, doi: 10.1007/s00500-012-0888-3

5.4 Fuzzy Nearest Neighbor Classification

Besides preprocessing techniques, there exist several approaches for improving the performance of nearest neighbor classification. One of the trends that have arisen in the last decades is the is the introduction of fuzzy sets theory [Zad65] within the mechanics of the nearest neighbor rule, giving birth to the field coined as fuzzy nearest neighbor classification.

The very first proposals were presented in 1983 [Jow83] and 1985 [KGG85]. Both of them define fuzzy membership to classes for every instance of the training set. Fuzzy membership are then considered as a *measure* of the confidence on the assignation of the instance to the class. Some clear examples of this would include having an instance with full membership to one class and zero to the rest - depicting a clear member to the former class - or having approximately the same membership to all the classes - meaning that the instance could be considered noisy or, at least, a very difficult example to classify -; however, the potential of fuzzy sets can still be used to extend further the nearest neighbor rule.

Once the literature on fuzzy nearest neighbor classification is analyzed thoroughly, it is possible to highlight three main places in which fuzzy sets have been introduced for modifying the nearest neighbor classifier:

- **Membership degree to a class**: Which included many different uses of fuzzy sets for representing the membership of the instances to classes, as is detailed above.
- **Similarity measure**: Fuzzy sets can be also incorporated to enhancement of the discriminative power of the training data through the modification of the similarity measure, that is, the function used to determine how different is each instance to the rest.
- **Decision rule**: In the nearest neighbor classifier, the final decision about the class of a test instance is given by a single majority voting process. This can also be upgraded by including fuzzy sets, as a way of developing more sophisticated voting schemes.

Studying the field, it is also possible to note that fuzzy nearest neighbor algorithms can be classified depending on the specific area of fuzzy sets considered for defining their basis. There have been defined many nearest neighbor algorithms based not only on fuzzy sets, but also in fuzzy rough sets [DCK07], intuitionistic fuzzy sets [Ata86], possibilistic theory [Zad78] and type-2 fuzzy sets [Men07].

After performing the study, we spotted the opportunity of bringing together some of these mechanisms to develop a new fuzzy nearest neighbor algorithm, in an effort to overcome the existing approaches. This work has enables us to develop two new approaches, IVF-kNN (Interval Valued Fuzzy k-Nearest Neighbors classifier), and EIVF-kNN (Evolutionary Interval Valued Fuzzy k-Nearest Neighbors classifier), being the latter an improvement of the former through the use of evolutionary algorithms.

The first model, IVF-kNN aims to improve the way in which memberships are managed by Fuzzy-KNN [KGG85], the classic fuzzy nearest neighbor classifier. In this model, although the memberships to classes are managed in a reasonable way, there can be still a lack of knowledge associated with the use of a single value to represent the membership. This problem can be avoided if several values, represented as type-2 fuzzy sets, are considered.

IVF-kNN thus introduces the use of interval valued type-2 fuzzy sets for avoiding the problem of using just a single value to represent the memberships, providing the general model with more flexibility. Interval valued fuzzy sets are also considered for managing the votes cast by each neighbor in the decision rule, as a new way of modeling the influence of each neighbor in the decision process.

The second model, EIVF-kNN, incorporates evolutionary algorithms to the fuzzy nearest neighbor framework. However, in this time evolutionary algorithms are not included as a tool for performing data reduction, but to determine the configuration of two key parameters of the IVF-kNN. This is a very important enhancement, specially if we consider that the first parameter is heavily attached to the process of computing the interval based membership approaches, whereas the second one adjusts the way in which the interval values votes are casted by each neighbor in the decision rule.

Following an approach somewhat similar to the rest of advanced nearest neighbor models presented before, a *wrapper* evolutionary model, conducted by the CHC evolutionary algorithm, has been selected to perform the search. The chromosomes of the algorithm encode different configurations of the two parameters, enabling the underlying IVF-kNN model to be adjusted as accurately as possible.

The journal articles associated to this part are:

- J. Derrac, S. García, F. Herrera, Fuzzy Nearest Neighbor Algorithms: Taxonomy, Experimental analysis and Prospects. **Submitted to Information Sciences**.
- J. Derrac, F. Chiclana, S. García, F. Herrera, EIVF-kNN: An Evolutionary Interval Valued Fuzzy K-Nearest Neighbors Classifier. **Submitted to IEEE Transactions on Fuzzy Sets and Systems**.

6. Discussion of results

The following subsections summarize and discuss the results obtained in each specific stage of the thesis.

6.1 Evolutionary Algorithms in Prototype Reduction

The use of evolutionary algorithms in prototype reduction has been characterized with the elaboration of the survey. The most essential techniques in prototype selection, prototype generation and training set generation have been analyzed thoroughly, focusing on the requirements for them to be applied over supervised learning problems.

Focusing on the field of prototype selection, it has been reviewed the most common way of representing the different subsets of prototypes selected during the search, and the different available methods to evaluate them. The best performing methods have been studied, which has allowed us to identify the most promising approaches for further developments.

Furthermore, some attention has been given to several hybrid methods combining prototype selection with other preprocessing techniques (such as feature selection or feature weighting). This underlying idea, the combination of several preprocessing techniques into a single algorithm, will be the foundation of some of the approaches that will be developed later.

Finally, it has been reviewed the application of evolutionary prototype reduction techniques to two challenging problems: The imbalance data sets problem and the scaling up problem. This analysis has shown how these problems can be successfully tackled with evolutionary algorithms, by adapting the algorithms and the evaluation of solutions to the specific problem in the former case [GH09], and by combining an advanced stratification strategy with an scalable evolutionary prototype selection algorithm in the latter [DGH10].

6.2 Coevolutionary Algorithms for Enhancing Nearest Neighbor Classification

Two advanced nearest neighbor classification models based on cooperative coevolution have been developed. Both are based in the incorporation of data preprocessing techniques to adapt the training data before performing the final classification rule of the nearest neighbor classifier.

The first model, IFS-CoCo, combines effectively instance selection, feature selection and dual selection. It has been presented as a robust evolutionary model, composed of three populations guided by the CHC algorithm. Its performance has been tested considering different evolutionary algorithms for instance selection, feature selection and dual selection. Existing hybrid evolutionary approaches have been also concluded.

The experimental study performed has shown that IFS-CoCo has a better performance that all the techniques selected for the comparison, overcoming also several classic techniques for instance selection and feature selection (non-evolutionary). Nonparametric statistical tests confirm our results, highlighting the differences between IFS-CoCo and the rest of algorithms as significant.

The second model, CIW-NN, features a combination of instance selection and weighting schemes. It also shows how cooperative coevolution can be used to joint effectively different evolutionary techniques (CHC and a steady-state genetic algorithm, in this case), with different codifications, into a single model.

The comparisons performed to test this second model are also favorable. Numerous methods of the state of the art in instance selection, feature weighting and instance weighting have been chosen, including SSMA and several composite techniques. Again, the comparison has been made in terms of accuracy and reduction of the training set, revealing that CIW-NN shows an outstanding performance in the task of enhancing the behavior of the nearest neighbor rule.

Both IFS-CoCo and CIW-NN are representative examples of the capabilities of cooperative coevolution, the former as a full data reduction model, and the latter as a hybrid model for data reduction and weighting. When configured properly, cooperative coevolutionary models are able to obtain outstanding results, improving by far the performance of standard approaches due to the successful combination of several techniques into a true cooperative algorithm.

6.3 Fuzzy Rough Sets and Evolutionary Algorithms in Nearest Neighbor Classification

Fuzzy rough sets have served as a suitable tool for performing data reduction in a nearest neighbor classification framework. Their capability of tackling both continuous and nominal data sets enables us to consider them within our general methodology.

Both fuzzy rough set based prototype selection and feature selection can be characterized by their competence on describing the indiscernibility relations present in the training data. This capacity, together with the good behavior of evolutionary algorithms in data reduction, has allowed us to develop two advanced nearest neighbor classifiers.

In our experimental studies, we have tested EIS-RFS considering evolutionary approaches for instance selection, feature selection and dual selection. The *Quickreduct* original version for feature selection has been also included. Finally, the joint application of evolutionary instance selection and the rough set based feature selection process one after another has also been included, as a way of determining if the combination of these techniques performed by EIS-RFS is better than the single application of both techniques.

The results shown that EIS-RFS overcomes significantly all the comparison algorithms with respect to classification accuracy, while acceptable reduction rates are obtained both with respect to instances and features. EIS-RFS also shows a satisfactory performance when large size data sets are considered. The experiments are completed with an study in which the enhancement to the k-NN classifier is depicted at different values of the k parameter. The conclusion of this last study is that the application of EIS-RFS is always beneficial for k-NN, although the benefits are greater the lower the k value is.

The second model, EFS-RPS also achieves satisfactory results when compared with evolutionary approaches for instance selection, feature selection and dual selection. This second model is also compared with *Quickreduct* and with the fuzzy rough set based prototype selection procedure, showing that it is able to effectively improve the results of the fuzzy rough set data reduction techniques in isolation. A final comparison is performed between EIS-RFS and EFS-RPS, showing that no hybrid algorithm is able to overcome the other.

The conclusions of this part are that both hybrid models have become two very robust approaches for improving nearest neighbor classification. This can be highlighted as a successful combination of fuzzy rough sets and evolutionary algorithms for data reduction, with the added benefit of having a computational cost lower than the expected for a pure evolutionary approach.

6.4 Fuzzy Nearest Neighbor Classification

Classical and recent approaches for fuzzy nearest neighbor classification have been thoroughly analyzed with the development of a review. As a result, we have proposed a new taxonomy, designed for characterizing every existing fuzzy nearest neighbor with respect to main category (fuzzy sets, type-2 fuzzy sets, possibilistic methods, intuitionistic fuzzy sets, fuzzy rough sets and preprocessing approaches via data reduction) and several key approaches including its dependence to the k parameter of the nearest neighbor rule or the way in which the different memberships are represented.

An extensive experimental study have been also carried out, comparing the performance of the most representative fuzzy nearest neighbor approaches. The best methods of each category have been highlighted, and a further comparison has been also performed including other representative crisp approaches for nearest neighbor classification.

The results of the comparison have shown the potential of fuzzy nearest neighbor classifiers and its good performance over general classification problems. Moreover, we have also remarked several key issues on fuzzy nearest neighbor classification, worth enough to be addressed in the future. These issues includes the extension of these techniques to handle nominal (discrete) and missing data, the development of new voting schemes, or the definition of advanced methods for estimating the fuzzy memberships of the instances to the different classes of the problem.

Based upon this last issue, we have developed a new fuzzy nearest neighbor classifier, IVF-kNN. This model uses type-2 fuzzy sets to represent the fuzzy memberships with an enhanced flexibility. Another model, EIVF-kNN, has been also developed. This second model extends further IVF-kNN by incorporating evolutionary algorithms to the process of adjusting the classifier to the available data, following a wrapper based approach.

Both models have shown an outstanding behavior when compared with representative approaches of the state of the art in fuzzy nearest neighbor classification. IVF-kNN outperforms both classical and recent approaches of the state of the art in the field, whereas EIVF-kNN is able to further overcome all of them, including IVF-kNN.

Thus, we can highlight EIVF-kNN as another competent advanced nearest neighbor classifier, obtained as a result of our research in fuzzy nearest neighbor classification. By incorporating both fuzzy sets and evolutionary algorithms into a single approach, we have developed a very competitive classifier, which has become the final contribution of this thesis to the field of nearest neighbor classification.

7. Concluding Remarks

We have addressed several issues pursuing a common objective: The development of advanced nearest neighbor classifiers. Our aim has been to design highly accurate classifiers, able to consume as little computational resources as possible in the final classification phase - both in the sense of reduction of the training set size, and in the sense of time spent by the classification rule.

This have been possible through the development of different techniques of data preprocessing. Some of them, like the inclusion of weights or the definition of class memberships are introduced towards an increase of the adaptability of the model to the training data, and ultimately to increase the classification accuracy of the models. On the other hand, data reduction techniques have also enhanced the models through the elimination of irrelevant, noisy and useless data (features or instances).

All these techniques have been applied by Soft Computing approaches. Soft Computing becomes the common element among all the works presented, whose evolution can be followed as each single objective is addressed. Firstly, evolutionary algorithms, one of the fundamental pillars of Soft Computing have been analyzed in the framework of data reduction for nearest neighbor classification. After understanding what are the requirements for their application and how they behave, and highlighting the best specific approaches, standard evolutionary techniques have given way to coevolution.

We have presented cooperative coevolution as a strategy for applying several related techniques over a single set of data, in a simultaneous way. Instead of employing coevolution in its traditional definition, as a tool for performing domain decomposition, our models show how different data preprocessing approaches can be performed jointly within a single evolutionary framework. As a result, it has been possible to obtain two very powerful classifiers, IFS-CoCo and CIW-NN, able to overcome most of the shortcomings of the nearest neighbor rule.

The second line of work has lead us to consider the inclusion of fuzzy rough sets. The fuzzy rough formalism has offered us with two very accurate, yet computationally not very costly, approaches for data reduction. Both techniques have been included into a evolutionary framework, producing as a result two orthogonal models: EIS-RFS (which features an evolutionary prototype selection algorithm guided by a fuzzy rough set based feature selection procedure) and EFS-RPS (an evolutionary feature selection algorithm conducted by a fuzzy rough set based prototype selection algorithm). Both models have shown an outstanding performance in its respective domains of competence.

Finally, fuzzy nearest neighbor algorithms have been studied as a step forward in the enhancement of the nearest neighbor rule. Probably, the most interesting characteristic of these algorithms is the capability of representing the degree of membership of the instances to each class of the domain, which means that a instance does not have to be assigned to just one class of the problems. Instead, with fuzzy nearest neighbor algorithms it is possible to mark dubious or borderline instances as members of two or more classes (with a given degree representing the *confidence* of this assignation), whereas instances with a clear classification can still keep a full degree of membership to their classes.

A full review on fuzzy nearest neighbor classification has been carried out, focused on characterizing the traits of the related techniques. Based upon all the lessons learned by performing this study, a new approach for fuzzy nearest neighbor classification, based on interval type-2 fuzzy sets, has been proposed. This technique, IVF-kNN, also features the capability of being optimized by evolutionary algorithms, which have given birth to its updated version, EIVF-kNN. This final model, hybrid between the evolutionary optimization techniques analyzed in the first stages of the thesis, and the fuzzy sets based knowledge representation employed in the last one, becomes our contribution to the state of the art of the field of fuzzy nearest neighbor classification.

Conclusiones

En esta tesis, hemos abordado varias cuestiones persiguiendo un objetivo común: El desarrollo de clasificadores del vecino ms cercano avanzados. Nuestro objetivo ha sido desarrollar clasificadores muy precisos, capaces de consumir tan pocos recursos como sea posible de cara a la clasificación final - tanto en tamaño del conjunto de entrenamiento como en tiempo empleado por la regla de clasificación.

Esto ha sido posible mediante el desarrollo de diferentes técnicas de preprocesamiento de datos. Algunas de ella, como la inclusión de pesos o la definición de pertenencias a clases han sido introducidas para obtener un incremento en la adaptación del modelo a los datos de entrenamiento, incrementando finalmente la precisión de los modelos de clasificación. Por otro lado, las técnicas de reducción de datos también han mejorado los modelos a través de la eliminación de datos irrelevantes, ruidosos e inútiles (tanto instancias como características).

Todas estas técnicas han sido aplicadas mediante propuestas de Computación Flexible. La Computación Flexible se ha convertido en el elemento común de todos los trabajos presentados, cuya evolución puede ser seguida a medida que cada objetivo propuesto es abordado. En primer lugar, los algoritmos evolutivos, uno de los pilares fundamentales de la Computación Flexible, han sido analizados en el ámbito de la reducción de datos para clasificación del vecino más cercano. Tras entender cuales eran los requisitos para su aplicación y cómo se comportaban, y destacar las propuestas más sobresalientes, las técnicas evolutivas estándar han dado paso a la coevolución.

Hemos presentado la coevolución cooperativa como estrategia para aplicar varias técnicas relacionadas sobre un único conjunto de datos, de forma simultánea. En lugar de emplear la coevolución siguiendo su definición tradiccional, como herramienta para realizar descomposiciones de dominio, nuestros modelos muestran como es posible aplicar de forma conjunta diferentes técnicas de reducción de datos en un solo marco de trabajo evolutivo. Como resultado, ha sido posible obtener dos clasificadores muy competitivos, IFS-CoCo y CIW-NN, capaces de salvar la mayoria de las desventajas de la regla del vecino más cercano.

La segunda línea de trabajo nos ha llevado a considerar la inclusión de conjuntos rugosos difusos. Este formalismo nos ha ofrecido dos propuestas muy precisas para reducción de datos, no costosas computacionalmente. Ambas técnicas han sido incluidas en un entorno evolutivo, produciendo como resultados dos modelos ortogonales: EIS-RFS (que incluye un algoritmo evolutivo de selección de prototipos guiado por un procedimiento de selección de características basado en conjuntos rugosos difusos) y EFS-RPS (un algoritmo evolutivo de selección de características conducido mediante un algoritmo de selección de prototipos basado en conjuntos rugosos difusos). Ambos modelos han mostrado un rendimiento sobresaliente en sus respectivos dominios.

Finalmente, se han estudiado los algoritmos del vecino más cercano difusos, como un paso adelante en la mejora de la regla del vecino más cercano. Probablemente, la característica más interesante de estos algoritmos es su capacidad de representar el grado de pertenencia de las instancias a cada clase del dominio del problema, lo que significa que una instancia no tiene porqué pertenecer a tan solo una clase. En lugar de eso, con los algoritmos del vecino más cercano difusos es posible marcar instancias dudosas o fronterizas como miembros de dos o más clases (con un grado de *confianza* con respecto a esta asignación), mientras que las instancias con una clasificación clara pueden aún mantener una pertenencia total a sus clases.

Se ha realizado una revisión completa de los algoritmos de clasificación del vecino más cercano difusos, centrada en caracterizar los rasgos más característicos de las técnicas relacionadas. Bsándonos en las lecciones aprendidas durante la realización del estudio, se ha presentado una nueva propuesta de clasificación del vecino más cercano difusa, basada en conjuntos difusos intervalares de tipo 2. Esta técnica, IVF-kNN, también dispone de la capacidad de ser optimizada mediante algoritmos evolutivos, lo que ha dado lugar al nacimiento de una versión extendida, EIVF-kNN. Este modelo final, híbrido entre las técnicas de optimización evolutiva analizadas al principio de la tesis y la representación de conocimiento basada en conjuntos difusos empleadas en la parte final, simboliza nuestra contribución al estado del arte en clasificación del vecino más cercano difusa.

8. Future Work

Besides the results reported in this thesis, numerous trends of work have been raised, some of them focused on exploiting the application of several advanced topics to further enhance the performance of the models proposed.

On the other hand, other future trends suggest the modification of the current techniques to adapt them for new challenging problems, some of them present in many real-world applications nowadays.

Incorporation of filter measures as fitness function in the evolutionary models Filter measures, as a way of evaluating the quality of a classifier, were defined traditionally in the field of feature selection [GE03]. *Filtering methods* are the counterpart of *wrapper approaches*, and differ from the latter in the fact that the quality of the selection is assessed through indirect measures (such as separability measures between classes), instead of estimating the performance through the classifier to be used.

The incorporation of filter measures in nearest neighbor models based on evolutionary algorithms would help to address one of their main drawbacks: The considerable cost of evaluating their fitness functions. However, this development is not straightforward, since a proper definition for assessing the quality of prototypes prior the classification phase would be necessary. This requirement, already met for feature selection (for example, by the fuzzy rough set based measure incorporated in *Quickreduct*), is still unsatisfied for prototype selection.

Upon developing a suitable filter measure for prototypes, a general definition could be applied to hybrid and advanced models. This would enable them to apply this new way of estimating the quality of the classifier without the necessity of building a new one every time a specific solution has to be evaluated.

Data reduction for general k nearest neighbor classifiers Most of the research performed on data reduction for nearest neighbor classification is focused on optimizing the 1-NN classifier. In the field of prototype selection, this is motivated by the fact that a value of $k = 1$ makes the k-NN rule more sensitive to noisy instances. This will provide the fitness function of evolutionary algorithms with the greatest possible sensitivity to noise during the reduction process, thus empowering their noise reduction capabilities. With a proper training phase, the results achieved would improve even those obtained through a search of the best global k parameter.

However, the determination of an appropriate set-up of the k parameter *locally* could be also incorporated to the search, as an advanced *locally weighted learning* procedure [AMS97]. In this context, *locally* means that certain areas of the data might be better classified by a large k value, whereas other areas could be better suited to a $k = 1$ set-up. This approach, (coined as *Multi-selection of instances* in [GPPR12]), can be regarded as the automatic determination of the number of votes that an instance casts during the voting process. In this manner, a k value can be learned per each instance selected, thus adapting the discriminative power of the instance in all the voting processes performed in its surroundings.

Fuzzy prototype selection and fuzzy prototype generation Fuzzy nearest neighbor algorithms can also be benefited by data reduction techniques. Regarding prototype selection [GDCH12], several fuzzy versions of the classical prototype selection techniques have been proposed (such as the Edited Fuzzy K-Nearest Neighbor Rule [YC98] or the Condensed Fuzzy k-Nearest Neighbor Rule [ZLZ11]). However, there is still a large potential for improving fuzzy nearest neighbor algorithms if advanced prototype selection techniques (including, for example, evolutionary algorithms) are considered.

Furthermore, prototype generation [TDGH12] poses a different way of focusing on the problem. The prototype generation techniques can also be applied to enhance fuzzy nearest neighbor classifiers without performing almost any modifications to the original models. However, the definition of the class membership of the prototypes in fuzzy nearest neighbor algorithms leads to a new way of obtaining the reduced training set: An optimal set of fuzzy prototypes, optimally positioned and with an optimum definition of their class memberships. Following this approach, it may be possible to design new fuzzy nearest neighbor classifiers even more accurate than the current state of the art.

Development of parallel, scalable and distributed models Advanced computational intelligence models based on metaheuristics (such as the ones proposed in this thesis) are prone to be improved with the use of parallelism. The employment of solutions based on this trend can be very advantageous for nearest neighbor classifiers based on Soft Computing techniques.

The benefits of such approaches are two-fold: On the first hand the underlying metaheuristics can take advantage of parallelism as a way to improve the quality of the solutions obtained [Alb05]. Advanced strategies, such as the decomposition of the search domain into several parts, where each one is governed by a search metaheuristic, can lead to an improvement in the performance of the search if the efforts of each piece are joined. This is closely related with they key elements of cooperative coevolution, already incorporated to two of the models proposed in this thesis.

On the second hand, scalable and distributed models can also be used to tackle large scale problems. For example, bigdata problems (applications dealing with huge amounts of data) are out of the reach of the current machine learning approaches [BBBE11]. However, they could be managed if proper scalable and distributed models would be considered when developing new models. A successful example of this kind of developments is the Apache Mahout project [1], which currently features several scalable machine learning libraries, able to process large collections of data.

Tackling imbalanced data and other difficult problems Besides standard classification problems, there are many other challenges in machine learning which are of great interest to the research community [YW06]. The unique capabilities of Soft Computing based nearest neighbor classifiers (high accuracy, great flexibility in the definition of the final training set, ...) gives them some opportunities of being useful when tackling some of these difficult problems.

For example, fuzzy nearest neighbor classification provides with a new way of dealing with the problem of imbalanced data [HG09, SWK09]. The prototypes' capability of learning different degrees of membership to each class makes them a very interesting tool for managing data in those cases where the importance of the classes is crucial. In an analogous way to cost-sensitive approaches [LFMTH12] (where misses in the minority classes are heavily penalized), fuzzy nearest neighbor approaches can also be sensitive to the classes distribution, and thus adapt the fuzzy memberships to reflect the importance of the prototypes of the minority classes in the decision rule.

[1] `http://mahout.apache.org/`

Chapter II

Publications: Published and Submitted Papers

1. Evolutionary Algorithms in Prototype Reduction

The journal paper associated to this part is:

1.1 A Survey on Evolutionary Instance Selection and Generation

- J. Derrac, S. García, F. Herrera, A Survey on Evolutionary Instance Selection and Generation. International Journal of Applied Metaheuristic Computing 1:1 (2010), 60-92, doi: 10.4018/jamc.2010102604
    - Status: **Published**.

A Survey on Evolutionary Instance Selection and Generation

Joaquín Derrac, University of Granada, Spain

Salvador García, University of Jaén, Spain

Francisco Herrera, University of Granada, Spain

ABSTRACT

The use of Evolutionary Algorithms to perform data reduction tasks has become an effective approach to improve the performance of data mining algorithms. Many proposals in the literature have shown that Evolutionary Algorithms obtain excellent results in their application as Instance Selection and Instance Generation procedures. The purpose of this article is to present a survey on the application of Evolutionary Algorithms to Instance Selection and Generation process. It will cover approaches applied to the enhancement of the nearest neighbor rule, as well as other approaches focused on the improvement of the models extracted by some well-known data mining algorithms. Furthermore, some proposals developed to tackle two emerging problems in data mining, Scaling Up and Imbalance Data Sets, also are reviewed.

Keywords: *Data Reduction, Evolutionary Algorithms, Imbalanced Data, Instance Generation, Instance Selection, Prototype Generation, Prototype Selection, Scaling Up, Training Set Selection*

1. INTRODUCTION

Data reduction (Pyle, 1999) is one of the data preprocessing tasks which can be applied in a data mining process. The main objective in data reduction is to reduce the original data by selecting its most representative information. This way, it is possible to avoid excessive storage and time complexity, improving the results obtained by any data mining application, ranging from predictive processes (classification, regression) to descriptive processes (clustering, extraction of association rules, subgroup discovery).

Data reduction processes can be performed in many ways, some of the more remarkable being:

- Selecting features (Liu & Motoda, 2007), reducing the number of columns in a data set. This process is known as Feature Selection.
- Making the feature values discrete (Liu et al., 2002), reducing the number of possible values of features. This process is known as attribute Discretization.

DOI: 10.4018/jamc.2010102604

- Generating new features (Guyon et al., 2006) which describe the data in a more suitable way. This process is known as Feature Extraction.
- Selecting instances (Liu & Motoda, 2001; Liu & Motoda, 2002), reducing the number of rows in a data set. This process is known as Instance Selection (IS)
- Generating new instances (Bezdek & Kuncheva, 2001; Lozano et al., 2006), which describes the initial data set by generating artificial examples. This process is known as Instance Generation (IG).

This article discusses a wide number of IS and IG proposals. They can be divided into two types of techniques depending on the goal followed by the reduction. If the set of selected or replaced instances will be used as the reference data to instance-based classification, then we refer to Prototype Selection (PS) and Prototype Generation (PG). On the other hand, if the set of instances obtained will be used as input or training set of any data mining algorithm for building a model, then we refer to Training Set Selection (TSS).

In spite of the differences between PS and PG (the first one finds suitable prototypes, while the second one generates them), both have been mainly employed to improve the same classifier, the Nearest Neighbor rule (Cover & Hart, 1967; see also Papadopoulos & Manolopoulos, 2004; Shakhnarovich et al., 2006). This predicts the class of a new prototype by computing a similarity measure (Cunningham, in press) between it and all prototypes from the training set. In the k-Nearest Neighbors classifier, k nearest prototypes vote to decide the class of the new instance to classify. This algorithm is the baseline of the instance based learning field (Aha et al., 1991).

On the other hand, TSS consists of the selection of reduced training sets to improve the efficiency and the results obtained by any data mining algorithm. It has been mainly applied to improve the performance of decision trees, neural networks and subgroup discovery techniques. Although there exists a wide number of TSS approaches, no IG work on TSS has been reported yet, until our knowledge.

In recent years, the data mining community has identified some challenging problems in the area (Yang & Wu, 2006). Two of these are the *Scaling Up Problem* and the *Imbalance Data Sets Problem*. They are closely related to the data reduction field.

The *Scaling Up Problem* (Provost & Kolluri, 1999; Domingo et al., 2002) appears when an overwhelming amount of data must be processed, overcoming the capabilities of the traditional data mining algorithms. The *Imbalance Data Sets Problem* (Chawla et al., 2004; Batista et al., 2004) appears when the distribution of the class in the training data is not balanced, thus the number of instances of some classes is too low. This distribution can cause several problems in the classification of examples which belong to the minority classes.

Evolutionary Algorithms (Eiben & Smith, 2003) are general-purpose search algorithms that use principles inspired by natural genetic populations to evolve solutions to problems. The basic idea is to maintain a population of chromosomes which represent plausible solutions to the problem and evolve over time through a process of competition and controlled variation.

Evolutionary Algorithms have been successfully used in different data mining (Freitas, 2002; Ghosh & Jain, 2005; Abraham et al., 2006) and data reduction (Cano et al., 2003; Oh et al., 2004) problems. Given that the IS problem can be defined as a combinatorial problem, Evolutionary Algorithms have been used to solve it with promising results (Ho et al., 2002; García et al., 2008); these applications of Evolutionary Algorithms to tackle IS problems are usually called EIS (Evolutionary Instance Selection) methods. Furthermore, Evolutionary Algorithms have shown interesting behavior in their application to IG due to it can be defined as a parameter optimization problem (Fernández & Isasi, 2004; Nanni & Lumini, 2008).

The aim of this article is to present a review on the use of Evolutionary Algorithms for PS, PG and TSS algorithms, called EIS-PS, EPG and EIS-TSS, respectively, giving their description and main characteristics. Several evolutionary proposals developed to tackle the *Scaling Up Problem* and the *Imbalance Data Sets Problem* will be included in the review.

This article is organized as follows: Section 2 presents the definitions of the techniques and problems which will be reviewed in the rest of the article. Section 3 presents an overall analysis of several EIS-PS methods. Section 4 reviews the EPG contributions presented in recent years. Section 5 deals with EIS-TSS methods and their application to tackle different data mining problems. Section 6, concludes the survey.

2. PRELIMINARIES

This section provides some preliminary concepts and definitions about the techniques and problems shown in the rest of this article. Firstly, we describe the main characteristics of IS and IG. Secondly, we present the *Scaling Up Problem* and the *Imbalance Data Sets Problem* for classification, with reference to their relevance to data reduction. Finally, we briefly review some common features of the Evolutionary Algorithms approaches applied to tackle these problems.

Instance Selection and Generation

IS is one of the main data reduction techniques. In IS, the aim 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 (Liu & Motoda, 2001). 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 $X_p = (X_{p1,}$ $X_{p2,}$, ..., $X_{pm,}$ 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 by *t* instances $X_{p.}$ Let $S \subset TR$ be the subset of selected samples that resulted from the execution of a IS algorithm, then we classify a new pattern from *TS* by a data mining algorithm acting over *S*. The whole data set is noted as *D* and it is composed of the union of *TR* and *TS*.

IS methods can be classified in two categories: PS methods and TSS methods. PS methods (Liu & Motoda, 2002) are IS methods which expect to find training sets offering best classification accuracy and reduction rates by using instance based classifiers which consider a certain similarity or distance measure. Recently, PS methods have increased in popularity within the data reduction field. Various approaches to PS algorithms have been proposed in the literature (see (Wilson & Martinez, 2000; Grochowski & Jankowski, 2004) for review). Figure 1 shows the basic steps of the PS process.

TSS methods are defined in a similar way. They are known as the application of IS methods over the training set used to build any predictive model (e.g. decision trees, neural networks …) Thus, TSS can be employed as a way to improve the behavior of predictive models, precision and interpretability (Riquelme et al., 2003). Figure 2 shows the basic steps of processing a decision tree (C4.5) on the TSS.

IG is another important technique in data reduction. It has been mainly applied to instance-based classifiers, thus we can focus on describing PG in depth. PG can be defined as the application of instance construction algorithms (Liu & Motoda, 2001) over a data set to improve the classification accuracy of a nearest neighbor classifier.

More specifically, PG can be defined as follows: Let X_p be an instance where $X_p = (X_{p1,}$ $X_{p2,}$, ..., $X_{pm,}$ 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

Figure 1. Prototype selection process

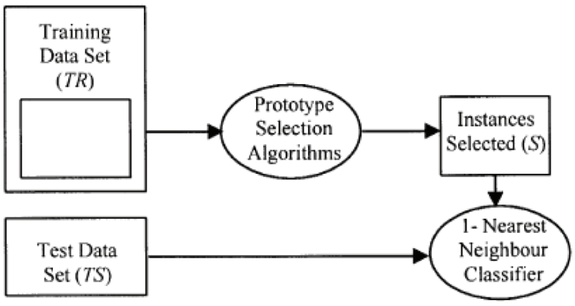
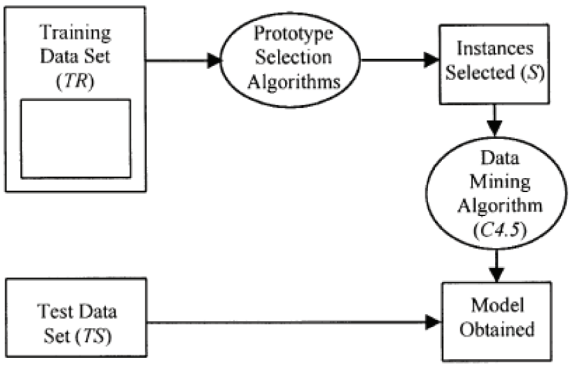


Figure 2. Training set selection process



a training set *TR* which consists of *N* instances X_p and a test set *TS* composed by *t* instances $X_{p.}$ The purpose of PG is to obtain a prototype generate set, which consists of *r; r <n,* prototypes, which are either selected or generated from the examples of X_p. The prototypes of the generated set are determined to represent efficiently the distributions of the classes and to discriminate well when used to classify the training objects. Their cardinality should be sufficiently small to reduce both the storage and evaluation time spent by a nearest neighbor classifier.

A wide number of PG methods have been designed in the specialized literature, ranging from traditional ones (Chang, 1974; Kohonen, 1990), to more modern approaches (Lozano et al., 2006). It is important to point out the fact that the research on the EPG field has started recently, being (Fernandez & Isasi, 2004) the first proposal in applying Evolutionary Algorithms to perform a PG task, in contrast to the research in evolutionary IS, where the first proposal was made in (Kuncheva, 1995).

The Scaling Up and the Imbalance Data Sets Problems

IS methods are considered a useful tool to perform data reduction tasks, obtaining interesting results. They have also been employed successfully to tackle two emergent challenges in data mining, the *Scaling Up Problem* and the *Imbalance Data SetsProblem* (Yang & Wu, 2006).

The *Scaling Up Problem* appears when the number of training samples increases beyond the capacity of the traditional data mining algorithms, harming their effectiveness and efficiency. Due to large size data sets, it produces excessive storage requirement, increases times complexity and affects to generalization accuracy. Usually, when the input data set size affects the execution of the algorithms, it is possible to face this situation with two different strategies:

- **Scaling up the algorithm:** Proposing faster and lower consumption algorithms that can face large size data sets. (Provost & Kolluri, 1999)
- **Scaling down the data set:** In this case, the attention is directed toward the data set. The idea consists of modifying the data set by means of reductions to make it adequate to the original algorithm (Liu & Motoda, 2002).

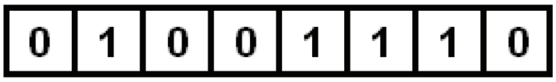
This problem has been recently addressed by many authors. An interesting example can be found in (Haro-García & García-Pedrajas, 2009), where a divide-and-conquer recursive approach to the problem is applied to very large problems, being able to match in accuracy and even improve on storage reduction the results of well-known standard IS algorithms with a very significant reduction in execution time. Another proposal, mainly adapted for use on evolutionary algorithms, is presented in (Cano et al., 2005), it will be described in the EIS-PS Section of this survey.

The *Imbalance Data Sets Problem* appears when the data contains many more examples of one class than the other and the less representative class represents the most interesting concept from the point of view of learning (Chawla et al., 2004). Imbalance in class distribution is pervasive in a variety of real-world applications, including but not limited to telecommunications (Tajbakhsh et al., 2009), web services, finance, ecology (Kubat et al., 1998), biology and medicine (Freitas et al., 2007).

Usually, in imbalanced classification problems, the instances are grouped into two types of classes: the majority or negative class, and the minority or positive class. The minority or positive class has more interest and it is also accompanied with a higher cost of misclassification. A standard classifier might ignore the importance of the minority class because its representation inside the data set is not strong enough. As a classic example, if the ratio of imbalance presented in the data is 1:100 (that is, there is one positive instance versus one hundred negatives), the error of ignoring this class is only 1%, so many classifiers could ignore it or not make any effort to learn an effective model for it.

Many approaches have been proposed to deal with the *Imbalance Data Sets Problem*. They can be divided into algorithmic approaches and data approaches. The first ones assume modifications in the operation of the algorithms, making them cost sensitive towards the minority class (Grzymala-Busse et al., 2005; Tajbakhsh et al., 2009). The data approaches modify the data distribution, conditioned on an evaluation function. Re-sampling of data could be done by means of under-sampling, by removing instances from the data (a process similar to IS) (Kubat & Matwin, 1997; Batista et al., 2004; Estabrooks et al., 2004), and over-sampling, by replicating or generating new minority examples (Chawla et al., 2002, Fernández et al., 2008).

Figure 3. A typical individual of an evolutionary IS algorithm



Basic Ideas on Evolutionary Algorithms for Instance Selection and Generation

A wide number of proposals presented on the use of Evolutionary Algorithms in IS and IG share some common characteristics regarding the two key concepts of any evolutionary algorithm: The representation of the population and the fitness function employed.

Representation: In most of the evolutionary IS algorithms, every member of the population encodes information about all the instances which are currently selected at each step of the search process.

This scheme is often used both in EIS-PS and EIS-TSS proposals. A binary representation is employed. Typically, every individual is defined as a binary string of length N, where each bit represents the current state of each instance of the training set (marked as '1' if the corresponding instance is currently selected, or '0' if not). Figure 3 shows a typical individual of an evolutionary IS algorithm.

On the other hand, in evolutionary IG algorithms every member of the population encodes information describing a new instance (or a new set of instances) to be generated. The concrete representation employed will vary depending on the codification of the problem data, although real coding is mostly preferred.

Fitness function: The majority of the evolutionary IS algorithms define a fitness function where two quality measures are employed: The accuracy of the results obtained by the subsequent data mining algorithm (e.g., a classifier), and the reduction rate achieved between the selected instances and the whole data set. Depending on the concrete method, this reduction rate can be computed by counting the number of instances selected, or employing more sophisticated methods, e.g. valuating the reduction as a measure of the interpretability of the tree obtained, when the evolutionary IS method is applied to improve the results of a decision tree algorithm.

In contrast, in evolutionary IG algorithms the only quality measure employed to define the fitness function is the accuracy obtained by employing the actual set of instances generated (typically, the accuracy obtained with a k-Nearest Neighbors classifier). The reduction rate is not usually employed because the number of instances generated is always small (usually it is fixed to a concrete value), thus every solution is expected to achieve a high reduction rate, and the search process can be focused only on the goal of increasing the accuracy of the classification process.

3. EVOLUTIONARY PROTOTYPE SELECTION

In this section, we will present the main contributions of EIS-PS appeared in the literature in recent years. Firstly, we give a snapshot on the *state of the art* in EIS-PS. Secondly, we describe in depth the characteristics of the most representative EIS-PS methods appeared. Thirdly, we show some EIS-PS proposals dealing with the *Scaling Up* and the *Imbalance Data Sets* problems. Finally, we conclude this section presenting some EIS-PS mixed approaches.

A Snapshot on Evolutionary Prototype Selection

The necessity of the Evolutionary Algorithms in PS is discussed in (Cano et al., 2003) where the authors differentiate between the selection based in heuristics (which appears in classic non-evolutionary PS algorithms, like for example CNN, IB3 or DROP described in (Wilson & Martinez, 2000)) and the selection developed by EIS-PS algorithms. EIS-PS presents a strategy that combines inner and boundary points. It does not tend to select instances depending on their a priori position in the search space (inner class or limit ones). EIS-PS selects the instances that increase the accuracy rates independently of their a priori position.

We will review the main contributions that have included or proposed an EIS-PS model in recent years. The first appearance of the application of an evolutionary algorithm to the PS problem can be found in (Kuncheva, 1995). Kuncheva applied a genetic algorithm to select a reference set for the k- Nearest Neighbors rule. Her genetic algorithm maps the training set onto a chromosome structure composed by genes, each one with two possible states (binary representation). The computed fitness function measures the error rate by application of the k-Nearest Neighbors rule. This genetic algorithm was improved in (Kuncheva & Bezdek, 1998; Ishibuchi & Nakashima, 1999).

At this point, all EIS-PS algorithms considered above adapt a classical genetic algorithm model to the PS problem. Later, a development of EIS-PS algorithms more conditioned to the problem is made. The first example of this can be found in (Sierra et al., 2001). In this article, an Estimation of Distribution Algorithm is used. Another example can be found in (Ho et al., 2002), where a genetic algorithm design for obtaining an optimal nearest neighbor classifier based on orthogonal arrays is proposed.

The technical term EIS-PS has been adopted by Cano et al. (2003), in which they analyze the behavior of different evolutionary algorithms, generational genetic algorithms (GGAs), steady-state genetic algorithms (SS-GAs), the CHC model (Eshelman, 1990) and Population Based Incremental Learning (PBIL) (Baluja, 1994) (which can be considered one of the basic Estimation of Distribution Algorithms). The fitness function used in these models combines two values: classification rate *clasRat* by using a 1-NN classifier and percentage reduction of prototypes of *S* with regards to *TR percRed*:

$$Fitness(S) = \alpha \cdot clasRat + (1-\alpha) \cdot perc\,\mathrm{Re}\,d$$

Where α is a weighting factor usually set to 0.5, and *perc_red* is defined as:

$$perc\,\mathrm{Re}\,d = 100 \cdot (|TR| - |S| / |TR|)$$

One of the newest approaches to EIS-PS employs memetic algorithms (Ong et al., 2007). These are heuristic searches in optimization problems that combine a population-based algorithm with a local search. The memetic algorithm employed by (García et al., 2008) incorporates an *ad hoc* local search specifically designed for optimizing the search in prototype selection problem with the aim of tackling the scaling up problem. Another recent proposal (Gil-Pita & Yao, 2008), is focused in the enhancing of the fitness function and the mutation and crossover operators when applied to PS problems.

Later research has gone further, focusing its interest on other topics apart from improving the design of EIS-PS algorithms. Several efforts have been focused on developing methods which will be able to tackle new challenges in data mining. A representative example can be found in (Cano et al., 2005), where an evolutionary method to tackle the *Scaling Up Problem* on the PS process is proposed. Another challenging problem addressed recently by using EIS-PS algorithms is the *Imbalanced Data Sets Problem* (García & Herrera, 2009).

A brief review will be done regarding mixed evolutionary approaches to IS with another data preprocessing technique, e.g. Feature Selection and Feature Weighting. Two different proposals will be reviewed, (Ros et al. 2008) being the first one. In this article, a hybrid genetic algorithm is applied to perform PS and Feature Selection simultaneously, trying to achieve three objectives simultaneously: Increasing the accuracy of the subsequent classification process, minimizing the dimensionality of the data (the number of features selected), and maximizing the number of instances selected.

The second proposal is an approach of data preprocessing with genetic algorithms in Case-Based Reasoning, which is described by the authors as an instance-based learning procedure (they also employ the k-Nearest Neighbors classifier). A basic genetic algorithm is conducted over a population of chromosomes which performs PS and Feature Weighting (which can be seen as a generalization of Feature Selection, where the weights are real valued between 0 and 1) simultaneously. This approach has been applied to two different scenarios: Customer classification and bankruptcy prediction modeling (Ahn et al., 2006; Ahn et al. 2009).

Overview on the EIS-PS Algorithms

In this subsection, we will describe in depth the characteristics of the most representative EIS-PS methods.

Generational Genetic Algorithm

Its basic idea is to maintain a population of chromosomes, which represent 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. The new chromosomes are created using genetic operators such as crossover and mutation.

In GGA, the selection mechanism produces a new population *P(t)* with copies of chromosomes in the old population *P(t-1)*. The number of copies received for each chromosome depends on its fitness; chromosomes with higher fitness have a greater chance of contributing copies to *P(t)*. Then, the crossover and mutation operators are applied to *P(t)*.

Algorithm 1 (See Figure 4) shows a basic pseudocode of GGA:

The GGA was the first scheme developed to perform EIS-PS processes. Although its results have been overcome by most of the subsequent proposals, it is still an important milestone in the field of PS.

Figure 4. Algorithm 1: Pseudocode of the GGA

```
Algorithm 1: GGA basic structure.
Input: A population.
Output: An optimized population.
1: Initialize population;
2: While Termination Criterion not satisfied do
3:     Evaluation of individual fitness;
4:     Formation of a gene pool (intermediate population) through selection
       mechanism;
5:     Recombination through crossover;
6:     Mutation operator;
7:     Replacement of the population;
8: end
```

Steady-State Genetic Algorithm

SSGA was firstly employed as an EIS-PS method in (Cano et al., 2003) In the SSGA usually one or two offspring are produced in each generation. Parents are selected to produce offspring and then a decision is made as to which individuals in the population will be selected for deletion in order to make room for the new offspring.

Algorithm 2 (See Figure 5) shows a basic pseudocode of SSGA:

In the construction of the SSGA, it is possible to select the replacement strategy (e.g., replacement of the worst, the oldest, or a randomly chosen individual) and the replacement condition (e.g., replacement if the new individual is better or unconditional replacement). A widely used combination is to replace the worst individual only if the new individual is better. Moreover, in (Goldberg & Deb, 1991), it is suggested that the deletion of the worst individuals can induce a high selective pressure, even when the parents are selected randomly.

This high selective pressure can help SSGA to improve the results of GGA in the performance of PS process. However, SSGA has been beaten also by most of the new proposals presented in recent years.

Population-Based Incremental Learning

PBIL (Baluja, 1994) is a specific Estimation of Distributions algorithm designed for binary search spaces. It attempts to explicitly maintain statistics about the search space to decide where to sample next.

The objective of the algorithm is to create a real valued probability vector V_p, which, when sampled, reveals high quality solution vectors with high probability. Initially, the values of V_p are set at 0.5. Sampling from this vector yields random solution vectors because the probability of generating a 1 or 0 for each gene is equal. As the search progresses, the values of V_p gradually shift to represent better solution vectors through the search process.

Algorithm 3 (See Figure 6) shows a basic pseudocode of PBIL algorithm.

The two basic search moves are performed in steps 6 and 8. In step 6, V_p is pushed toward S_{Best}. *LR* is the learning rate, which specifies how close the movement to S_{Best} is. In step 8, V_p is pushed far away from S_{Worse} . LR_{neg} is the negative learning rate, which specifies how far away the steps are from the worst solution.

PBIL can be seen as one of the most representative evolutionary proposals for performing PS process, because it is almost the only example of EIS-PS method which its search process is

Figure 5. Algorithm 2: Pseudocode of the SSGA

```
Algorithm 2: SSGA basic structure.
Input: A population.
Output: An optimized population.
1: Initialize population;
2: While Termination Criterion not satisfied do
3:     Select two parents from the population;
4:     Create one/two offspring using crossover and mutation;
5:     Evaluate the offspring with the Fitness function;
6:     Select one/two individuals in the population, which may be replaced by the
       offspring;
7:     Decide if this/these individuals will be replaced;
8: end
```

Figure 6. Algorithm 3: Pseudocode of the PBIL algorithm

```
Algorithm 3: PBIL algorithm basic structure.
Input: $V_p$.
Output: Optimized $V_p$.
1:  Initialize $V_p$;
2:  While Termination Criterion not satisfied do
3:      Generate a fixed number of samples based upon the probabilities specified
        in $V_p$;
4:      Select the best sample, $S_{Best}$, and the worst sample, $S_{Worst}$;
5:      Foreach I in $V_p$ do
6:          $V_p[i] = V_p[i] * (1-LR) + S_{Best}[i] * LR$;
7:          If $S_{Best}[i] != S_{Worst}[i]$ then
8:              $V_p[i] = V_p[i] * (1-LR_{neg}) + S_{Best}[i] * LR_{neg}$;
9:          end
10:     end
11: end
```

not based in a genetic algorithm. In general, it obtained good results when compared with genetic-based methods in the study carried in (Cano et al., 2003).

The Evolutionary Model CHC

CHC algorithm (Eshelman, 1990) is a binary-coded genetic algorithm which involves the combination of a selection strategy with a very high selective pressure, and several components inducing a strong diversity. CHC is a robust evolutionary algorithm, which should often offer promising results in several search problems.

The four main components of the algorithm are:

- **An elitist selection:** To compose a new generation, the best individuals among parents and offspring are selected.
- **A highly disruptive crossover:** HUX, which crosses over exactly half of the non-matching alleles.
- **An incest prevention mechanism**, which only allows to cross over those pairs of individuals which has a Hamming distance higher than a difference threshold. This threshold is decreased, as time goes by, to help the population to converge.
- **A restart process**, which is applied when the population has converged (when the threshold has dropped to zero). It generates a new population by randomly flipping a percentage (usually a 35%) of the bits of the old population individuals.

Algorithm 4 (See Figure 7) shows a basic pseudocode of CHC algorithm.

In the study carried out by (Cano et al., 2003), the CHC algorithm was selected as the best EIS-PS strategy, being able to outperform all the remaining methods of the study (the evolutionary and the non-evolutionary ones).

An interesting conclusion derived from that study was that the key feature of the CHC algorithm is its ability to select the most representative instances independently of their position in the search space, satisfying both the objectives of high accuracy and reduction rates. Due to this fact, CHC has been widely used as a baseline method to perform many evolutionary IS tasks. Some examples are shown in the next sections of this survey.

Figure 7. Algorithm 4: Pseudocode of the CHC algorithm

```
Algorithm 4: CHC algorithm basic structure.
Input: A population.
Output: An optimized population.
1:  Initialize population;
2:  While Termination Criterion not satisfied do
3:      Select candidates from the population;
4:      Generate offspring by crossing parents;
5:      Evaluate the offspring with the fitness function;
6:      Select the individuals of the new population
7:      If population not changed then
8:          Decrease threshold;
9:      end
10:     If threshold<0 then
11:         Restart population and reinitialize threshold;
12:     end
13: end
```

Intelligent Genetic Algorithm

Ho et al. (2002) propose the Intelligent Genetic Algorithm (IGA) based on Orthogonal experimental design used for PS and Feature Selection. Besides its initial definition, it can also be applied as a PS method only, without changing its initial objectives of increasing accuracy and reduction rates on the training data.

IGA is a GGA that incorporates an Intelligent Crossover operator. It builds an orthogonal array from two parents of chromosomes and searches within the array for the two best individuals according to the fitness function. It takes about $2^{\log_2 (\gamma-1)}$ fitness evaluations to perform an Intelligent Crossover operation, where γ is the number of bits that differ between both parents. Note that the application of Intelligent Crossover operator to large-size chromosomes (resulting chromosomes from large size data sets) could consume a high number of evaluations.

Algorithm 5 (See Figure 8) shows a basic pseudocode of IGA.

As their authors concluded, the employment of the Intelligent Crossover operator allows IGA to be superior to conventional genetic algorithms when applied to problems where the solution space is large and complex, e.g. when it is composed of high dimensional overlapping patterns. Thus, it is a good EIS-PS method to apply when facing medium and large sized data sets.

Steady-State Memetic Algorithm

The steady-state memetic algorithm (SSMA) was proposed in (García et al., 2008) to cover a drawback of the conventional EIS-PS methods that had appeared before: their lack of convergence when facing large problems.

SSMA makes use of a local search or meme specifically developed for this prototype selection problem. This interweaving of the global and local search phases allows the two to influence each other; i.e. SSGA chooses good starting points, and local search provides an accurate representation of that region of the domain. This local search scheme assigns a probability value to each chromosome generated by crossover and mutation, C_{new}:

$$P_{LS} = \begin{cases} 1 & if\ Fitness(C_{new})\ is\ better\ than\ Fitness(C_{worst}) \\ 0.625 & otherwise \end{cases}$$

Figure 8. Algorithm 5. Pseudocode of the IGA

```
Algorithm 5: IGA basic structure.
Input: A population.
Output: An optimized population.
1: Initialize population;
2: While Termination Criterion not satisfied do
3:     Evaluate all individuals using the fitness function;
4:     Use rank selection to select individuals to make a new population;
5:     Randomly cross some individuals by employing IC crossover;
6:     Select one/two individuals in the population, which may be replaced by the
       offspring;
7:     Apply the conventional bit inverse mutation operator to the population;
8: end
```

Figure 9. Algorithm 6: Pseudocode of the SSMA

```
Algorithm 6: SSMA basic structure.
Input: A population.
Output: An optimized population.
1: Initialize population;
2: While Termination Criterion not satisfied do
3:     Use binary tournament to select two parents;
4:     Apply crossover operator to create offspring ($Off_1$, $Off_2$);
5:     Evaluate $Off_1$ and $Off_2$;
6:     Foreach $Off_i$ do
7:         Invoke Adaptive-PLS-mechanism to obtain $PLS_i$
8:         If $u(0,1) < PLS_i$ then
9:             Perform meme optimization for $Off_i$;
10:        end
11:    end
12:    Employ standard replacement for $Off_1$ and $Off_2$;
13: end
```

Algorithm 6 (See Figure 9) shows a basic pseudocode of the SSMA.

Where u(0,1) is a value in a uniform distribution u[0,1] and the standard replacement means that a the worst individual is replaced only if the new individual is better.

The *Adaptive-PLS-mechanism* is an adaptive fitness-based method. A description of the *Adaptive-PLS-mechanism* and the meme specifically developed for the prototype selection task can be found in (García et al., 2008).

The meme optimization mechanism is a local search specifically designed for the PS problem. It tries to improve the initial chromosome by generating its neighbors by unselecting one of its current selected prototypes. These neighbors are evaluated by a special fitness function which is able to consume only partial evaluations, saving computational resources for the whole evolutionary process.

The objective of the meme optimization mechanism is adjusted dynamically in the execution of the SSMA. Every time a certain number of evaluations have been spent, the accuracy and reduction rates achieved by the best chromosome of the population are registered. If the classification accuracy has not increased, then the meme optimization starts an *improving accuracy* stage, where only better results in accuracy are accepted through the local search. On the other hand, if the reduction rate has not increased, then the meme optimization starts an *Avoiding premature convergence* stage, where the local search accepts worse solutions in order to improve the diversity of the population.

As their authors concluded, the SSMA presents a good reduction rate and computational time. In fact, it is able to outperform the classical PS algorithms, when the accuracy and reduction rates are considered. When compared to other EIS-PS methods, SSMA is able to outperform or equal them, being particularly useful as the size of the databases increases.

Genetic Algorithm Based on Mean Square Error, Clustered Crossover and Fast Smart Mutation

A new genetic algorithm (concretely a GGA model) was proposed in (Gil-Pita & Yao, 2008) as an EIS-PS model. This algorithm (no name was provided by their authors) included the definition of a novel mean square error based fitness function, a novel clustered crossover technique, and a fast smart mutation scheme.

The fitness function employed was based on a mean square error measure:

$$F = \frac{1}{NC}\sum_{n=1}^{N}\sum_{i=1}^{C}(\frac{K_n[i]}{K} - d_n[i])^2$$

Where N is the number of training patterns, C is the number of classes, K is the number of nearest neighbors employed, K_n is the number of K nearest neighbors belonging to class *i*, and d_n is 1 when the desired output for the instance n is the class I, and 0 when not. As their authors stated, the error surface defined by this function is smoother than those obtained using counting estimator based functions, making easier the obtaining of its local minimum.

The clustering crossover operator firstly performs a k-means clustering process over the chromosomes, extracting the centroids of all the clusters found (the number of clusters is established randomly). Then, the centroids are employed with a classical random point cross operator to generate the individuals of the new generation.

The fast smart mutation procedure computes the effect of the change of one bit of the chromosome over its fitness value, testing all the possibilities. The change which produces the better fitness value is accepted as the result of the mutation operator. It is applied to every individual of the population. At the end of its application, the fitness value of the individuals is actualized, starting then a new generation of the evolutionary process.

Algorithm 7 (See Figure 10) shows a basic pseudocode of the algorithm.

The results obtained by the authors in the experimental study which was carried out suggested that the joint use of the three proposed methods could be quite interesting in the case of not very large training sets.

EIS-PS Proposals for the Scaling Up and Imbalanced Data Problems

In this subsection, we will analyze some EIS-PS proposals dealing with the *Scaling Up* and the *Imbalance Data Sets* problems.

Stratification of EIS-PS to Tackle the Scaling Up Problem

Cano et al. (2005) proposed a method to tackle the *Scaling Up Problem* in EIS-PS. The method presented consists of a stratified strategy which divides the initial data set into disjoint strata with equal class distribution. The number of strata chosen will determine their size, depending on the size of the data set. Using the proper number of strata the stratified method is able to significantly reduce the training set, avoiding the drawbacks of the *Scaling Up Problem*.

Figure 10. Algorithm 7: Pseudocode of the genetic algorithm based on mean square error, clustering crossover and fast smart mutation

```
Algorithm 7: Genetic algorithm based on mean square error, clustering crossover
and fast smart mutation
-----------------------------------------------------------------------------
Input: A population.
Output: An optimized population.
-----------------------------------------------------------------------------
1: Initialize population;
2: While Termination Criterion not satisfied do
3:    | Evaluate population;
4:    | Apply clustering crossover (only one time for each 10 generations);
5:    | Apply Fast Smart Mutation operator;
6: end
```

Figure 11 shows the basic steps of the process.

Following the stratified strategy, the initial data set *D* is divided into *t* disjoint sets D_j, strata of equal size, D_1, D_2, …, D_t maintaining class distribution within each subset. Then, PS algorithms will be applied to each D_j obtaining a selected subset DS_j. In this way, the subsets *TR* and *TS* will be obtained as follows:

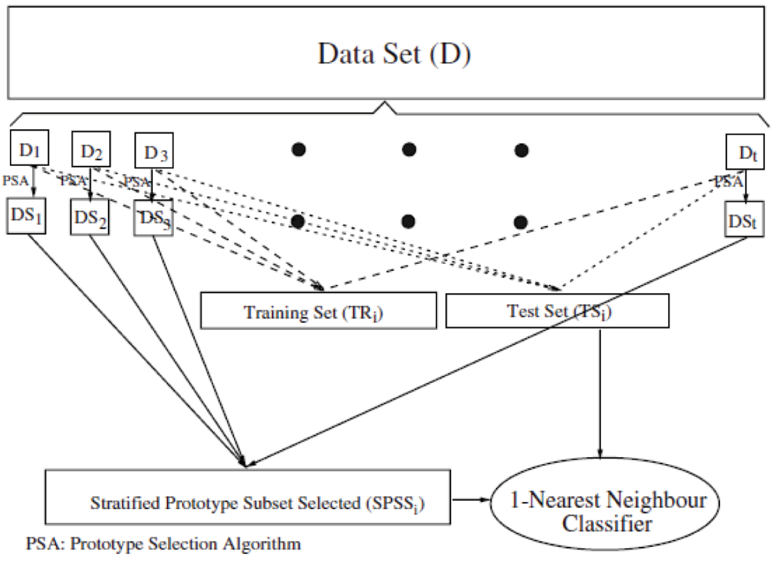
$$TR = \bigcup_{j \in J} D_j, J \subset \{1,2,..,t\} \quad TS = D - TR$$

And the Stratified Prototype Subset Selected (SPSS) is defined as:

$$SPSS = \bigcup_{j \in J} DS_j, J \subset \{1,2,..,t\}$$

The nearest neighbor classifier is then evaluated using as training data the *SPSS* set, and the *TS* set as test data. Thus the classifica-

Figure 11. Structure of the stratification process in EIS-PS


Data Set (D)
D1
D2
D3
Dt
PSA
PSA
PSA
PSA
DS1
DS2
DS3
DSt
Training Set (TRi)
Test Set (TSi)
Stratified Prototype Subset Selected (SPSSi)
1-Nearest Neighbour Classifier
PSA: Prototype Selection Algorithm

tion process can be performed on higher size data sets, avoiding the usual drawbacks and still achieving acceptable results.

The concluding remarks of the study were that a proper choice in the number of strata makes it possible to decrease significantly execution time and resources consumption, maintaining the EIS-PS algorithm's behavior in accuracy and reduction rates. Also, the CHC was selected as the best EIS-PS algorithm when employed within the evolutionary stratified PS process.

EIS-PS Algorithms on Imbalanced Data Problems

In (García & Herrera, in press), a complete study of EUS (Evolutionary Under-Sampling) algorithms is carried out. A set of EUS methods is proposed, which take into consideration the nature of the problem and use different fitness functions to obtain a good trade-off between balance of distribution of classes and performance.

Eight different algorithms compose the set of methods proposed in the study. Furthermore every method shares the same basic structure, which is developed by using the CHC algorithms as an evolutionary model. There are three characteristics that differentiate them:

- The objective that they pursue.
    - Aiming for an optimal balancing of data without loss of effectiveness in classification accuracy. EUS models that follow this tendency will be called Evolutionary Balancing Under-Sampling.
    - Aiming for an optimal power of classification without taking into account the balancing of data, considering the latter as a sub-objective that may be an implicit process. EUS models that follow this tendency will be called Evolutionary Under-Sampling guided by Classification Measures.
- The way that they do the selection of instances.
    - If the selection scheme proceeds over any kind of instance, then it is called Global Selection. That is, the chromosome contains the state of all instances belonging to the training data set and removals of minority class instances (those belonging to positive class) are allowed.
    - If the selection scheme only proceeds over majority class instances then it is called Majority Selection. In this case, the chromosome saves the state of instances that belong to the negative class and a removal of a positive or minority class instance is not allowed.
- The accuracy measure used for its fitness function.
    - Geometric Mean methods, if the geometric mean is used as accuracy measure. Geometric mean was first employed in (Barandela et al., 2003} and is defined as:

$$GM = \sqrt{a^{+} \cdot a^{-}}$$

Where a^{+} denotes accuracy in examples belonging to the minority class and a^{-} denotes accuracy in examples belonging to the majority class.

- Area Under the ROC Curve methods, if the Area Under the Curve measure is used instead the geometric mean. Area Under the ROC Curve (Bradley, 1997) can measure the efficacy of various classifiers simultaneously, employing the True Positive and False Positive rates of a classification process.

The eight methods were tested with several imbalanced datasets, and the results obtained were contrasted by using non-parametric statistical procedures. The main conclusions drawn from the study were:

- PS algorithms must not be used for handling imbalanced problems. They are prone to gain global performance by eliminating examples belonging to the minority class considered as noisy examples.
- During the evolutionary under-sampling process, the employment of a majority selection mechanism helps to obtain more accurate subsets of instances than the use of global selection. However, the latter mechanism is necessary to achieve the highest reduction rates.
- Data sets with a low imbalance ratio should be faced with Evolutionary Under-Sampling guided by Classification Measures models, and should use in particular the model with a global mechanism of selection and evaluation through the geometric mean measure.
- Data sets with a high imbalance ratio should be faced with Evolutionary Balancing Under-Sampling models, and should use in particular the model with a majority selection mechanism and evaluation through the geometric mean measure.

Mixed EIS-PS Approaches

In recent years, some proposals have appeared performing not only the PS process with evolutionary algorithms, but also performing another data preparation process simultaneously. This subsection will review two of the most remarkable approaches.

Ros et al. (2008) proposed a hybrid genetic algorithm (HGA) which performs IS and Feature Selection simultaneously. Its objectives are to increase the accuracy of the k- Nearest Neighbors over the reference set, to minimize the number of features selected (reducing the reference data) and to maximize the number of instances selected (to retain the most information possible without harming the classification accuracy).

The HGA is divided into three phases:

- A genetic algorithm is applied in the first phase. It includes a sophisticated selection scheme and some mechanisms to manage diversity and elitism (including an archive population and a dynamic analysis of the diversity of the population).
- By employing a histogram of the frequency with which each feature has been selected, a feature selection process is carried out, in order to simplify the problem.
- The genetic algorithm is applied again over the population. Also, some of the children generated by each generation are tuned by using local search procedures.

Despite the contradictory objectives in the number of instances and features selected, HGA is able to perform a dual IS and FS process with success, being a suitable evolutionary method to perform data reduction tasks.

A second mixed approach, GOCBR (Global Optimization of feature weighting and instance selection using genetic algorithms for Case Based Reasoning) was proposed in (Ahn et al., 2006; Ahn et al., 2009). This proposal performs a simultaneous IS and Feature Weighting process in the framework of a Case Based Reasoning system.

The search process of GOCBR consists of the application of a genetic algorithm with the common genetic operators (selection, crossover and mutation). Their individuals employ a binary representation, encoding the weights of the features by employing 14 bits to each one, and encoding the IS information in the second part of the chromosome by employing the usual binary scheme. Its fitness function measures only the accuracy obtained by employing the reference set defined by the chromosome to classify the train data in a k- Nearest Neighbors classifier.

GOCBR system has been applied successfully by the authors to various problems, such as customer classification or bankruptcy prediction modeling. Also, it is remarkable that is the only evolutionary method known of which performs a simultaneous IS and Feature Weighting process, until our knowledge.

4. EVOLUTIONARY PROTOTYPE GENERATION

In this section, we will present the main contributions of EPG appeared in the literature in recent years. In the first subsection, we give a snapshot on the *state of the art* in EPG. In the last subsection, we describe in depth the characteristics of the most representative EPG methods appeared.

A Snapshot on Evolutionary Prototype Generation

In recent years, the research efforts in the design of new PG techniques based on Evolutionary Algorithms have started to offer some interesting approaches. All of them still employ the NN rule as a reference classifier to measure the classification accuracy of the prototypes generated.

Usually, the prototypes are encoding as members of the population of the evolutionary algorithm employed to carry out the evolutionary process. A real codification scheme is used to represent them; each of its components has a concrete value for a concrete feature of the problem.

In this section, we will review the main contributions that have proposed an EPG model. The first contribution is (Fernandez & Isasi, 2004), where an evolutionary algorithm based on a two-dimensional grid is proposed, named Evolutionary Nearest Prototype Classifier. It defines a traditional evolutionary process to prepare the prototypes for its use on a 1-NN classifier, by employing a wide number of evolutionary operators.

The next two proposals are based on the Particle Swarm Optimization (PSO) Scheme (Kennedy et al., 2001). This technique is based on a set of potential solutions (particles) which evolves to find the global optimum of a real-valued function (fitness function) defined in a given space (search space). Particles represent the complete solution to the problem and move in the search space using both local information (the particle memory) and neighbor information (the knowledge of neighbor particles).

In (Nanni & Lumini, 2008) a PSO based PG method is proposed (no name is provided for the algorithm). It can be seen as a Pittsburgh-based model, because all the components of the solution (in this case, the prototypes generated) are encoded in a single particle. The method performs a PSO search process where a reduced set of prototypes is generated to finally perform a 1-NN classification process.

In (Cervantes et al., 2007; Cervantes et al, in press), inspired by the results of (Cervantes et al., 2005), an Adaptive Michigan PSO model for PG is proposed. This model is described as a Michigan approach because every particle contains only a component of the solution, thus the complete solution is built by joining the selected particles of the swarm. This approach also improves the traditional PSO scheme, because the online generation and destruction of particles is allowed in the search process.

Finally, the last proposal which will be reviewed, (Garain, 2008), is based on the Clonar Selection Algorithm (Castro & Zuben, 2002), a representation of an Artificial Immune System model (Dasgupta, 1998). Clonar Selection Algorithms are inspired by the behavior of the immune system when performing an immune response to an antigenic stimulus. It advocates the idea that only those cells that recognize the antigens proliferate, thus being selected against those which do not. This idea is employed to develop a PG system which is able to generate suitable prototypes to perform a 1-NN classification process.

Overview on the EIG Algorithms

In this subsection, we will describe in depth the characteristics of the most representative EIG methods.

Evolutionary Nearest Prototype Classifier

In (Fernandez & Isasi, 2004), an EPG method is proposed. It employs as a basic structure a two-dimensional matrix, where each row is associated with a prototype of the whole classifier, and each column is associated with a class to define regions where the prototypes are mapped.

On its initialization, the algorithm only defines one prototype. Then a whole evolutionary process starts: It carries out sequentially a set of evolutionary operations with the aim of generating a robust set of prototypes which will be able to correctly generalize the instances of the train set. The evolutionary operators defined are:

- **Mutation:** This operator is used to label each prototype with the most heavily populated class in each of its own regions.
- **Reproduction:** The reproduction operator function is to introduce new prototypes into the classifier, splitting the instances assigned to a prototype into two sets, where the second set is assigned to a new prototype.
- **Fight:** This operator allows prototypes to exchange their assigned instances. The fight can be performed in a cooperative or a competitive way, and it is ruled by the quality of the prototypes involved.
- **Movement:** The movement operator reallocates a prototype on the centroid of its assigned instances.
- **Die:** The current prototypes have a chance of being erased from the matrix, which is inversely proportional to its quality.

In the whole process, the quality of each prototype is defined by the number of prototypes which it has currently assigned and its classification accuracy. When the evolutionary process is finished, the generated set of prototypes is employed to classify the tests set, by means of the 1-NN classifier.

Algorithm 8 (See Figure 12) shows a basic pseudocode of Evolutionary Nearest Prototype Classifier algorithm.

The Evolutionary Nearest Prototype Classifier algorithm has shown good overall results, when compared against well-known classical methods in PG. Moreover, the results obtained when employing the prototypes generated with a 1-NN classifier were very competitive when compared with many classical instance-based classifiers as C4.5, Naïve Bayes or PART.

Particle Swarm Optimization for Prototype Generation

In (Nanni & Lumini, 2008) a PG method based on PSO is presented. This method defines the particles of the swarm as sets of a fixed number of prototypes, which are modified as the particle is moved in the search space.

Figure 12. Algorithm 8: Pseudocode of the Evolutionary Nearest Prototype Classifier algorithm

```
Algorithm 8: Evolutionary Nearest Prototype Classifier algorithm.
Input: Training set.
Output: A set of prototypes.
1: Initialization;
2: While Termination Criterion not satisfied do
3:     Collect information describing the prototypes;
4:     Mutation phase;
5:     Reproduction phase;
6:     Fight phase;
7:     Movement phase;
8:     Die phase;
9: end
```

The usual operators of PSO are employed by this proposal. The representation of each particle consists of a vector of length S= K · *M* given by the concatenation of *K* prototypes (dealing with *M*-dimensional data). The fitness function employed is defined as the classification error of the set of *K* prototypes over the training data.

Several runs (*N*) of the PSO process are carried out before finishing the PG stage (the authors recommend N=5). Each execution gives as result a reference set of *K* prototypes, being the result of the PG stage a collection of *N* reference sets. To classify a test instance, it is evaluated by each of the *N* reference sets, obtaining the final output as the result of a majority vote above all the reference sets. Thus, the proposed model can be seen as an ensemble of PSO-based classifiers.

Algorithm 9 (See Figure 13) shows a basic pseudocode of PSO-based PG algorithm.

The employment of the ensemble structure allows this proposal to obtain high accuracy rates. Furthermore, the authors suggested some ways to improve the model (like the employment of feature weighting methods). This proposal confirms that PSO is a very suitable model to perform PG processes.

An Adaptive Michigan Approach PSO for Nearest Prototype Classification

In (Cervantes et al., 2007; Cervantes et al., in press), an Adaptive Michigan Approach PSO is proposed. Its particles encode a prototype, each one being the generated train data represented as the whole particle swarm. This method does not have a fixed number of particles. On the contrary, some new operations are defined to allow the PSO search procedure to increase or decrease dynamically the number of particles.

The algorithm employs two different fitness functions: The global fitness function, defined by the standard classification accuracy on a 1-NN classifier, which is used to find the best swarm over the whole PSO procedure; and a local fitness function valued in each particle, defined by using the number of prototypes correctly classified and misclassified by itself. This secondary fitness function is used to evaluate the quality of each particle, in order to judge if it must be erased from the swarm, or if it can be employed as a parent of a new particle.

When the whole PSO process has finished, a cleaning process is carried out on the best swarm found. This swarm is the final output of the algorithm.

Algorithm 10 (See Figure 14) shows a basic pseudocode of the Adaptive Michigan PSO algorithm:

Figure 13. Algorithm 9: Pseudocode of the PSO-Based PG algorithm

```
Algorithm 9: PSO-based PG algorithm.
Input: Training set and test set.
Output: Classification output for the test set.
1: Initialization;
2: For j=1:N do
3:     PG(j)= PSO-PG(Train set_j);
4: end
5: Foreach x_i in test set do
6:     For j=1:N do
7:         PartialOutput(j)= Classify(x_i, PG(j))
8:     end
9:     Output(x_i)= VoteRule(PartialOutput);
10: end
```

This second PSO-based approach is focused in obtaining very high accuracy rates. The employment of a Michigan representation allows the definition of a local fitness function, which helps the method to find quickly suitable solutions in the search space.

Prototype Reduction Using an Artificial Immune Model

In (Garain, 2008), a PG based on a Clonar Selection algorithm is proposed. This model is composed of an immune memory which stores in its cells the best antigens found in the search process.

The Clonar Selection algorithm is initialized by representing the training instances as antigens, and choosing one antigen from each class to fill the immune memory. Then the search process starts. The first stage consists of a Hyper-mutation process. For each antigen on the training set, the most stimulating antigen in the immune memory is selected. The measure of stimulation is based on how close both antigens are (by means of Hamming or Euclidean Distance). The selected antigen of the memory is used as a parent for the Hyper-mutation process, which generates its offspring. Then, a Resource Allocation procedure is called, which balances the total number of clones present in the system by giving half of the resources to the clones of the class of the current antigen. The other half is equally divided among clones of other classes.

While the classification accuracy is improved by the generation of clones, further Mutation processes (and Resource Allocation procedures) are carried out on the surviving clones. This Mutation produces a lower number of clones which depends on the stimulation value of each parent clone. When no improvement is achieved, the best clone found is inserted into the immune memory, performing a replacement with the worst antigen present. Then a new generation starts.

Finally, when the algorithm meets a global termination criterion, the antigens contained in the immune memory are employed as the training set to classify the test instances, by using the 1-NN classifier.

Algorithm 11 (See Figure 15) shows a basic pseudocode of PG-Clonar Selection algorithm:

Figure 14. Algorithm 10: Pseudocode of the AMPSO algorithm

```
Algorithm 10: Adaptive Michigan PSO algorithm.
Input: Training set.
Output: A set of prototypes (best swarm).
1:  Initialize swarm. Dimension of particles equals number of attributes;
2:  Insert N particles of each class into the training patterns;
3:  While iterations < MAX and accuracyRate < 100% do
4:      Check for particle reproduction and deletion;
5:      Foreach particle do
6:          Calculate local fitness;
7:          Calculate its next position;
8:      End
9:      Move the particles;
10:     Assign classes to the training patterns using the nearest particle;
11:     Evaluate the swarm classification accuracy;
12: End
13: Delete, from the best swarm found so far, the particles that can be removed
without a reduction in the classification accuracy.
```

The Clonar Selection Algorithm is a new method for PG based on a field of the Evolutionary Computation which has started to grown recently: The immune systems. Although it has high storage requirements to allocate the clones generated, it is a first example of a new technique which can obtain promising results with further research.

5. EVOLUTIONARY TRAINING SET SELECTION

In this section, we will present the main contributions of EIS-TSS appeared in the literature in recent years. In the first subsection, we give a snapshot on the *state of the art* in EIS-TSS. The next subsections will analyze the main approaches of EIS-TSS in decision trees, neural networks and subgroup discovery, respectively.

A Snapshot on Evolutionary Training Set Selection

The advances in EIS-TSS in recent years have been directed towards improving the results of some well-known data mining algorithms, being principally focused on the enhancement of the performance of decision trees, neural networks and subgroup discovery.

A wide range of these proposals have been inspired by the good results obtained by EIS-PS in the task of improving the performance of instance-based classifiers. Thus, some of the EIS-TSS methods which will be presented in this section will share some components with EIS-PS proposed before, adapting its principles to tackle the IS problems over other data mining algorithms.

Firstly, we will review some approaches of EIS-TSS applied to the construction of decision trees with the well-known C4.5 algorithm (Quinlan, 1993). A first application of IS to improve the construction of decision trees can be found in (Cano et al., 2003), where the results of the IS conducted by four evolutionary proposals are applied to extract reduced training sets in the construction of decision trees. Another proposal (Wu & Olaffson, 2006) also presented a genetic algorithm based IS process to improve the construction of decision trees, but they focused its effort on improving, not only the accuracy of the model and the reduction of the number of instances in the training set, but also in the interpretability and size of the trees obtained.

Later, Cano et al. (2007) presented two proposals of stratification to further improve the trees extracted by the C4.5 algorithm. The aim of both proposals was to improve the accuracy and interpretability of the trees extracted

Figure 15. Algorithm 11: Pseudocode of the PG-Clonar Selection algorithm

Algorithm 11: PG- Clonar Selection algorithm.

Input: Training set.
Output: A set of prototypes (immune memory).

```
1:  Initialization;
2: While Termination Criterion not satisfied do
3:      Proliferation I (Hyper-mutation);
4:      Resource Allocation;
5:      While resources left do
6:          Proliferation II (Mutation);
7:          Resource Allocation;
8:      end
9:      Insert best antigen into immune memory;
10: end
```

by means of stratification of the training data, trying to maintain a good trade-off between both quality measures.

As a last application of EIS-TSS in decision trees, we will review a proposal to improve the performance of C4.5 over imbalanced data sets (i.e. dealing with the *Imbalance Data Sets Problem* in the construction of decision trees) (García & Herrera, 2008). In this contribution it is shown how an evolutionary undersampling method is able to increase the accuracy of the decision tress in the classification of both majority and minority classes, employing the geometric mean accuracy measure.

Two approaches of EIS-TSS applied to neural networks will be analyzed. The first approach is (Ishibuchi et al., 2001), where a basic genetic algorithm is applied to perform both IS and Feature Selection processes to improve the results of standard three layered neural networks in classification.

A second approach to perform EIS-TSS on neural networks is presented in (Kim, 2006). This approach proposes the use of a standard genetic algorithm to perform a weight adjustment on the connections between the layers of a feed-forward neural network and an IS process over the instances which are employed to train the net.

Finally, the review of two proposals of EIS-TSS applied to subgroup discovery will be covered to close this section. The first proposal of this subsection (Cano, Herrera, Lozano & García, 2008) is an enhancement of the CN2-SD algorithm to increase its efficiency over large size datasets, by employing TSS techniques.

The second proposal (Cano et al., 2008) presents two stratification strategies to increase the presence of examples from minority classes in large size data sets with imbalanced data. The benefits shown in this study from the application of stratification includes the enhancement of Apriori-SD in its application to large size problems, and the improvement in the quality of the groups discovered over the minority classes of the problem analyzed.

EIS-TSS in Decision Trees

Cano et al. (2003) performed a complete study of the use of Evolutionary Algorithms to perform IS tasks. Although this work has been analyzed above, due to the number of EIS-PS which were presented in the study, it is important to note that a second experimental study was carried out employing the same IS algorithms as EIS-TSS methods to improve the trees built by C4.5.

The results obtained in the TSS part of the study were similar to the ones reached in the PS part: Evolutionary Algorithms based IS method were able to equal or outperform non-evolutionary methods, maintaining or increasing the accuracy of the trees obtained and increasing the reduction rates obtained by measuring the number of instances selected.

Wu & Olaffson (2006) performed a wide analysis of the application of IS to the induction of decision tress. They proposed a genetic algorithm to conduct the IS process, employing an integer codification in its chromosomes. Each individual is composed of a set of integer values, where each one represents one instance of the training set. Thus the selected instances of each individual are those whose integer identifier forms part of the chromosome.

The genetic algorithm search process employs a set of usual genetic operators: roulette wheel selection, crossover operator (which interchanges members of each set of instances defined by the parents) and mutation operator (which randomly replaces an instance by another one not selected).

The fitness function is a measure of the accuracy of the tree which can be induced by the chromosome, *S*, and its size. It is defined as follows:

$$Fitness(S) = -\log(e(\psi(S))) - a \cdot \log(\frac{size(\psi(S))}{K}),\ a > 1$$

Where e is an estimator of the error rate of the decision tree $\psi(S)$, K is an upper bound on the size of the tree, and a is a weighting factor.

The final tree is obtained by merging the instances selected at least one time with a fixed number of the best chromosomes in the population. Then the tree can be employed to classify new test instances.

In addition, a study of some relevant parameters is presented along with the results of the algorithm. A discussion of two additional measures, the Average Leaf Ratio (a measure of the number of instances in each leaf node of the tree), and the Instance Entropy of the training data, show some interesting conclusions, the most remarkable being:

- Genetic algorithm based IS is able to reduce the tree sizes with minimal loss in prediction accuracy
- The best results are obtained when the Average Leaf Ratio of the final model is higher
- Genetic algorithm based IS works better on low entropy data sets. Higher values of entropy make the construction of more complex decision trees necessary
- IS can be employed as a replacement for the traditional tree pruning techniques because it is able to outperform them when they are compared in terms of accuracy and tree size

A third remarkable proposal of EIS-TSS can be found in (Cano et al., 2007), where the use of stratification is proposed to tackle the *Scaling Up* problem on the induction of decision trees. The aim of the study was to perform the extraction of classification rules from large size data by keeping a good tradeoff between the precision and the interpretability of the model generated. To accomplish its objective, the authors present a stratified strategy (similar as the employed in (Cano et al., 2005).

To conduct the EIS-TSS process, the CHC algorithm is used. Moreover, two different fitness functions are employed; they are based on the usual fitness function used for EIS-PS:

$$Fitness(TSS) = \alpha \cdot clasPer + (1-\alpha) \cdot redPer$$

Where *clasPer* denotes the percentage of correctly classified objects from TR using only TSS to find the nearest neighbor or to extract a C4.5 model, depending on the concrete fitness function used. *redPer* denotes the reduction rate between TR and TSS.

- **Reduction-precision fitness function:** This fitness function uses the 1-Nearest Neighbor classifier for measuring the classification rate
- **Interpretability-precision fitness function:** This fitness function extracts a model with C4.5 to compute the classification performance of TSS

The two fitness definitions were tested with CHC and the stratification strategy. Several TSS methods were included in the experimental framework, to compare the performance of the proposed models against them. Finally, the results obtained were contrasted by using non-parametric statistical procedures.

The main conclusions reached in the study were as follows:

- The evolutionary stratified IS offers the best model size, maintaining an acceptable accuracy. It produces the smallest set of rules, with the minimal number of rules and the smallest number of antecedents per rule.
- The stratified CHC model with Interpretability-Precision fitness function allows us to obtain models with high test accuracy rates, similar to C4.5, but with the advantage that the size of the models are reduced considerably.
- The predictive model extraction by means of evolutionary stratified training set selection (with the CHC model and any of the fitness function presented) presents a good tradeoff between accuracy and interpretability. Thus, a very good scaling up behavior is observed, which allows us to obtain good results when the size of data set grows.

The last EIS-TSS proposal to improve the construction of decision trees which will be reviewed in this survey deals with the *Imbalance Data Sets* problem.

In (García & Herrera, in press) a new method is presented to deal with imbalanced data by performing the TSS process. The aim of the method is to improve the classification accuracy obtained by C4.5 when it is used on imbalanced data sets.

The proposed approach uses the same representation as the basic EIS-TSS methods. C4.5 is used to extract a model in order to compute the accuracy of the training set selected. The accuracy rates obtained by employing this model to classify the examples of the majority and minority classes are used to compute the geometric mean metric, which is used as fitness function.

Figure 16 shows the basic stages of the EUS process in EIS-TSS.

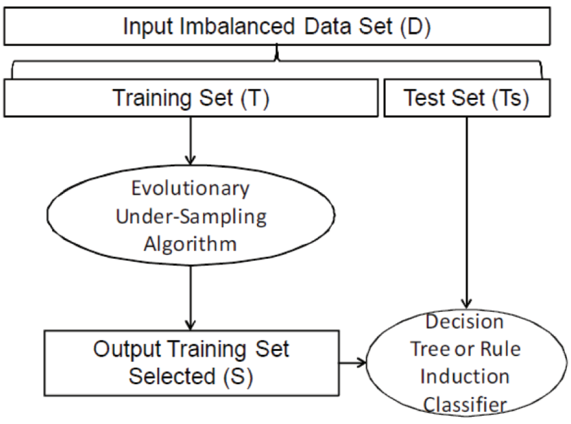
Although C4.5, in its standard definition, incorporates a pruning mechanism to avoid overfitting, the inclusion of the induction tree process within an evolutionary cycle can direct the resulting tree to an optimal model for training data, losing the generalization ability. To avoid this drawback, a simple and effective mechanism is incorporated. It consists of providing a higher weight for the classification costs of the instances that are not included in TSS than to the instances that are. Therefore, the reduction ability of the selected subset is encouraged, allowing the proposed approach to avoid overfitting in the construction of the models.

To test the performance of the proposal, it was compared to a wide number of well-known re-sampling algorithms, including OSS (Kubat & Matwin, 1997), NCL (Laurikkala, 2001) and SMOTE (Chawla et al., 2002), among others. The results obtained were contrasted by using non-parametric statistical procedures, finally showing that the proposed approach is able to outperform, or, at least, to behave similarly to every method in the comparison with respect to the accuracy of the models obtained, it obtains very accurate trees with a low number of rules or leafs. Thus, the proposed approach is confirmed as a very accurate method, which is able to increase the interpretability of the models obtained.

EIS-TSS in Neural Networks

A first application of the use of EIS-TSS on neural networks can be found in (Ishibuchi et al., 2001). The aim of this proposal was to find an optimal subset of instances and features by employing a GGA.

Figure 16. The EUS process in EIS-TSS


Input Imbalanced Data Set (D)
Training Set (T)
Test Set (Ts)
Evolutionary Under-Sampling Algorithm
Output Training Set Selected (S)
Decision Tree or Rule Induction Classifier

The GGA designed to carry out the TSS task employed the usual representation of the solutions (binary coded, employing the first part of the chromosome to code the instances which are currently selected, and the second part to code the features), and a standard set of genetic operators: Random selection of parents, uniform crossover and bit flipping mutation (biased to decrease the number of instances selected). The fitness function defined by this proposal was:

$$Fitness = W_{Perf} \cdot Perf - W_f \left|F\right| - W_P \cdot \left|P\right|$$

Where *Perf* is a measure of the accuracy of a NN classifier when applied to the training data by using as a reference set the current subset selected by the chromosome, *F* and *P* are respectively the number of features and instances selected, and W_{Perf}, W_F and W_P are user-defined weights.

When the evolutionary process was finished, the final subset selected was employed to train a standard three layered neural network, being finally validated with a test data set to test its generalization capability.

Although the concrete EIS-TSS method employed has been outperformed by other EIS-TSS and EIS-PS proposals, this approach can still be considered an important milestone in the field because it was the first application of EIS-TSS to improve the performance of neural networks.

A more modern approach is proposed in (Kim, 2006). In this proposal, a GGA is also employed to optimize the performance of a three layered neural network, in the framework of an application for financial forecasting. The individuals of the genetic algorithm encode information about the instances selected and about the adjustment of the weights of the neural network, employing binary coding in both cases. Although an EIS-TSS process is performed, the genetic algorithm does not have as its objective the maintenance of a good reduction rate. Instead, the fitness function is defined as the classification accuracy of the neural network defined by the chromosome, instead of the usual application of the neural network as a baseline classifier.

The experimental study carried out concluded that, in the context of the financial forecasting problem selected, the application of IS to improve the performance of neural networks outperformed the classical proposals of weights adjustment with genetic algorithms, highlighting the benefits of using EIS-TSS to improve the quality of the training process of neural networks.

EIS-TSS in Subgroup Discovery

In (Cano, Herrera, Lozano & García, 2008), a proposal to improve the performance of the CN2-SD algorithm for subgroup discovery in the evaluation of large size data sets is presented. Although CN2-SD is based on a divide and conquer strategy, it has to face the *Scaling Up* problem. To avoid it, the use of TSS algorithms is proposed for scaling down the data sets before the subgroup discovery task.

The study of the application of TSS algorithms, and the experiments that were carried out, was divided into two parts:

- In the first part, the effect of TSS on the subgroups discovered with CN2-SD in small data sets is studied. The objective is to analyze if the TSS process affects the descriptive qualitative measures of the subgroups (coverage, support, confidence, significance, unusualness, completeness, size and number of antecedents).

The basic IS methods applied to the TSS process were CNN (Hart, 1968), IB2, IB3 (Kibbler & Aha, 1987), DROP3 (Wilson & Martinez, 1997) ICF (Brightom & Mellish, 2002) and EIS-CHC. These methods were applied by using the stratification proposed in (Cano et al., 2005).

The results of this first part of the study were contrasted to a complete set of parametrical and non-parametrical statistical tests. Their application revealed that the use of TSS did not negatively affect the quality indexes of the subgroup discovered. Also the measures on size and number of antecedents were improved, showing that TSS algorithms were able to discover smaller and more interpretable sets of subgroups.

- In the second part, a TSS process is combined with CN2-SD to test its behavior in large size data sets. As basic IS methods, IB2 and EIS-CHC were selected because they were the IS algorithms with the smallest subsets selected in their application to the large size data sets.

The main conclusion of this part was that the combination of the highest reduction rates of IB2 and EIS-CHC with CN2-SD makes it possible to perform a SD task on large size data sets. In particular, EIS-CHC is recommended because it shows very good results in most of the qualitative measures tested, when employed in combination with CN2-SD.

As a final conclusion of the study, the authors stated that, thanks to the application of TSS methods, CN2-SD can be executed on large data set sizes pre-processed, maintaining and improving the quality of the subgroups discovered.

A second application of EIS-TSS for subgroup discovery can be found in (Cano et al., 2008). There, a different application of the stratified strategy presented in (Cano et al., 2005) was proposed: The employment of two modified strategies of stratification to increase the presence of minority classes. The aim of the proposal was to allow a subgroup discovery algorithm, Apriori-SD (Kavsek & Lavrac, 2006), to avoid the *Scalability problem* and to face a large data set without harming its accuracy due to a poor treatment of imbalanced data.

The data set used on the experiment was the KDD Cup'99. Firstly, it is shown that the Apriori-SD could not handle the KDD Cup'99 problem because of its expensive computational cost in time. Then the TSS methods are applied to tackle the problem. ENN (Hart, 1968), IB3 (Kibbler & Aha, 1987), and EIS-CHC (Cano et al., 2005) were proposed as baseline IS methods to be used.

To preserve the number of instances of the minority classes, two different strategies of stratification were proposed:

- **Instance selection in all classes:** The instances of the majority classes are assigned randomly over the strata created. Then the whole minority classes are added to each strata. After the IS process is carried out, the subsets selected are reunited, removing duplicities.

The employment of this strategy showed a severe drawback: Its application to every TSS subset obtained after the reunion of the instances selected from the strata decreased significantly the number of instances present from the minority classes. Thus, the minority classes were not sufficiently represented for a proper subgroup discovery task, due to use of the stratified IS.

- **Instance selection in majority classes:** The selection process is applied without the minority ones, just to the majority classes. The instances which belong to the minority classes were added to the TSS subset after the reunion of the subsets selected, and then the subgroup discovery tasks were carried out.

In this case, the instances which appear in the TSS selected were the most representative of the majority classes and all the instances belonging to the minority ones. Thus the IS process was able to reduce the initial data set without affecting the presence of instances from the minority classes, making the subgroup discovery process possible in those classes.

Both strategies were tested in combination with the selected IS methods, using confidence and support as qualitative measures. The conclusions of the experiment were that the combination of the TSS algorithms with stratification allows us to extract subgroups for most of the classes, including most of the minority ones, with high levels of confidence and support measures. Furthermore, the use of the *instance selection in majority classes* strategy of stratification was recommended to perform this task.

6. CONCLUSION

This article presents a review of PS, TSS and PG techniques performed by evolutionary algorithms. A wide number of algorithms and proposals of the *state-of-the-art* have been discussed, showing that the research in these fields have produced numerous advances in recent years to improve the quality of Instance Selection and Generation techniques in Data Mining.

Furthermore, this survey has considered the use of Evolutionary Algorithms to tackle two important issues in Data Mining: *The Scaling up Problem* and the *Imbalance Data Sets Problem*. These proposals have provided a way to improve the results obtained over large sized and imbalanced data in such fields as supervised classification and subgroup discovery, being a clear example of how Evolutionary Algorithms can be a useful tool in order to overcome these challenging problems.

ACKNOWLEDGMENT

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

REFERENCES

Abraham, A., Grosan, C., & Ramos, V. (Eds.). (2006). *Swarm intelligence in data mining*. Berlin, Germany: Springer-Verlag.

Aha, D. W., Kibler, D., & Albert, M. K. (1991). Instance-based learning algorithms. *Machine Learning*, *6*, 37–66.

Ahn, H., & Kim, K. (2009). Bankruptcy prediction modeling with hybrid case-based reasoning and genetic algorithms approach. *Applied Soft Computing*, *9*, 599–607. doi:10.1016/j.asoc.2008.08.002

Ahn, H., Kim, K., & Han, I. (2006). Hybrid genetic algorithms and case-based reasoning systems for customer classification. *Expert Systems: International Journal of Knowledge Engineering and Neural Networks*, *23*(3), 127–144. doi:10.1111/j.1468-0394.2006.00329.x

Baluja, S. (1994). *Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning* (Tech. Rep. CMU-CS-94-163). Pittsburgh, PA: Carnegie Mellon University.

Barandela, R., Sánchez, J. S., García, V., & Rangel, E. (2003). Strategies for learning in class imbalance problems. *Pattern Recognition*, *36*(3), 849–851. doi:10.1016/S0031-3203(02)00257-1

Batista, G. E. A. P. A., Prati, R. C., & Monard, M. C. (2004). A study of the behavior of several methods for balancing machine learning training data. *ACM Special Interest Group on Knowledge Discovery and Data Mining . SIGKDD Explorations*, *6*(1), 20–29. doi:10.1145/1007730.1007735

Bezdek, J. C., & Kuncheva, L. I. (2001). Nearest prototype classifier designs: An experimental study. *International Journal of Intelligent Systems*, *16*(12), 1445–1473. doi:10.1002/int.1068

Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, *30*(7), 1145–1159. doi:10.1016/S0031-3203(96)00142-2

Brightom, H., & Mellish, C. (2002). Advances in instance selection for instance based learning algorithms. *Data Mining and Knowledge Discovery*, *6*, 153–172. doi:10.1023/A:1014043630878

Cano, J. R., García, S., & Herrera, F. (2008). Subgroup discovery in large size data sets preprocessed using stratified instance selection for increasing the presence of minority classes. *Pattern Recognition Letters*, *29*, 2156–2164. doi:10.1016/j.patrec.2008.08.001

Cano, J. R., Herrera, F., & Lozano, M. (2003). Using evolutionary algorithms as instance selection for data reduction in KDD: An experimental study. *IEEE Transactions on Evolutionary Computation*, *7*, 561–575. doi:10.1109/TEVC.2003.819265

Cano, J. R., Herrera, F., & Lozano, M. (2005). Stratification for scaling up evolutionary prototype selection. *Pattern Recognition Letters*, *26*, 953–963. doi:10.1016/j.patrec.2004.09.043

Cano, J. R., Herrera, F., & Lozano, M. (2007). Evolutionary stratified training set selection for extracting classification rules with trade-off precision-interpretability. *Data & Knowledge Engineering*, *60*, 90–100. doi:10.1016/j.datak.2006.01.008

Cano, J. R., Herrera, F., Lozano, M., & García, S. (2008). Making CN2-SD subgroup discovery algorithm scalable to large size data sets using instance selection. *Expert Systems with Applications*, *35*, 1949–1965. doi:10.1016/j.eswa.2007.08.083

Castro, L. N., & Zuben, F. V. J. (2002). Learning and optimization using the clonal selection principle. *IEEE Transactions on Evolutionary Computation*, *6*, 239–251. doi:10.1109/TEVC.2002.1011539

Cervantes, A., Galván, I., & Isasi, P. (2007). An adaptive michigan approach PSO for nearest prototype classification. In *Nature Inspired Problem-Solving Methods in Knowledge Engineering* (LNCS 4528, pp. 287-296).

Cervantes, A., Galván, I., & Isasi, P. (in press). AMPSO: A new particle swarm method for nearest neighborhood classification. *IEEE Transactions on Systems, Man and Cybernetics, part B*.

Cervantes, A., Isasi, P., & Galván, I. (2005). A comparison between the pittsburgh and michigan approaches for the binary pso algorithm. In *Proceedings of the 2005 IEEE Congress on Evolucionary Computation,* Munchen, Germany (pp. 290-297).

Chang, C.-L. (1974). Finding prototypes for nearest neighbor classifiers. *IEEE Transactions on Computers*, *23*(11), 1179–1184. doi:10.1109/T-C.1974.223827

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, *16*, 321–357.

Chawla, N. V., Japkowicz, N., & Kotcz, A. (2004). Editorial: Special issue on learning from imbalanced data sets. *ACM Special Interest Group on Knowledge Discovery and Data Mining . SIGKDD Explorations*, *6*(1), 1–6. doi:10.1145/1007730.1007733

Cover, T. M., & Hart, P. E. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, *13*, 21–27. doi:10.1109/TIT.1967.1053964

Cunningham, P. (in press). A taxonomy of similarity mechanisms for case-based reasoning. *IEEE Transactions on Knowledge and Data Engineering*.

Dasgupta, D. (Ed.). (1998). *Artificial immune systems and their applications*. Berlin, Germany: Springer Verlag.

Domingo, C., Gavalda, R., & Watanabe, O. (2002). Adaptive sampling methods for scaling up knowledge discovery algorithms. *Data Mining and Knowledge Discovery*, *6*(2), 131–152. doi:10.1023/A:1014091514039

Eiben, A. E., & Smith, J. E. (2003). *Introduction to evolutionary computing*. Berlin, Germany: Springer Verlag.

Eshelman, L. J. (1991). The CHC adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination. In G. Rawlins (Ed.), *Foundations of genetic algorithms and classifier systems* (pp. 265-283). San Mateo, CA: Morgan Kaufmann.

Estabrooks, A., Jo, T., & Japkowicz, N. (2004). A multiple resampling method for learning from imbalanced data sets. *Computational Intelligence*, *20*(1), 18–36. doi:10.1111/j.0824-7935.2004.t01-1-00228.x

Fernández, A., García, S., del Jesus, M. J., & Herrera, F. (2008). A study of the behaviour of linguistic fuzzy rule based classification systems in the framework of imbalanced data-sets. *Fuzzy Sets and Systems*, *159*(18), 2378–2398. doi:10.1016/j.fss.2007.12.023

Fernández, F., & Isasi, P. (2004). Evolutionary design of nearest prototype classifiers. *Journal of Heuristics*, *10*, 431–454. doi:10.1023/B:HEUR.0000034715.70386.5b

Freitas, A. A. (2002). *Data mining and knowledge discovery with evolutionary algorithms*. New York: Springer-Verlag.

Freitas, A. A., da Costa Pereira, A., & Brazdil, P. (2007). Cost-sensitive decision trees applied to medical data. In *Data Warehousing and Knowledge Discovery* (LNCS 4654, pp. 303-312).

Garain, U. (2008). Prototype reduction using an artificial immune model. *Pattern Analysis & Applications*, *11*, 353–363. doi:10.1007/s10044-008-0106-1

García, S., Cano, J. R., & Herrera, F. (2008). A memetic algorithm for evolutionary prototype selection: A scaling up approach. *Pattern Recognition*, *41*(8), 2693–2709. doi:10.1016/j.patcog.2008.02.006

García, S., & Herrera, F. (in press). Evolutionary under-sampling for classification with imbalanced data sets: Proposals and taxonomy. *Evolutionary Computation*.

Ghosh, A., & Jain, L. C. (Eds.). (2005). *Evolutionary computation in data mining*. Berlin, Germany: SpringerVerlag.

Ghosh, A., & Jain, L. C. (Eds.). (2005). *Evolutionary computation in data mining*. Berlin, Germany: Springer Verlag.

Gil-Pita, R., & Yao, X. (2008). Evolving edited k-nearest neighbor classifiers. *International Journal of Neural Systems*, *18*(6), 1–9. doi:10.1142/S0129065708001725

Goldberg, D. E., & Deb, K. (1991). A comparative analysis of selection schemes used in genetic algorithms. In G. Rawlins (Ed.), *Foundations of genetic algorithms and classifier systems* (pp. 69-93). San Mateo, CA: Morgan Kaufmann.

Gómez-Ballester, E., Micó, L., & Oncina, J. (2006). Some approaches to improve tree-based nearest neighbour search algorithms. *Pattern Recognition*, *39*(2), 171–179. doi:10.1016/j.patcog.2005.06.007

Grochowski, M., & Jankowski, N. (2004). Comparison of instance selection algorithms II. Results and comments. In *Proceedings of Artificial Intelligence and Soft Computing - ICAISC 2004* (LNCS 3070, pp. 580-585).

Grzymala-Busse, J. W., Stefanowski, J., & Wilk, S. (2005). A comparison of two approaches to data mining from imbalanced data. *Journal of Intelligent Manufacturing*, *16*, 565–573. doi:10.1007/s10845-005-4362-2

Guyon, I., Gunn, S., Nikravesh, M., & Zadeh, L. (Eds.). (2006). *Feature extraction*. Heidelberg, Germany: Springer.

Haro-García, A., & García-Pedrajas, N. (2009). A divide-and-conquer recursive approach for scaling up instance selection algorithms. *Data Mining and Knowledge Discovery*, *18*, 392–418. doi:10.1007/s10618-008-0121-2

Hart, P. E. (1968). The condesed nearest neighbour rule. *IEEE Transactions on Information Theory*, *18*(3), 431–433.

Ho, S.-Y., Liu, C. C., & Liu, S. (2002). Design of an optimal nearest neighbor classifier using an intelligent genetic algorithm. *Pattern Recognition Letters*, *23*(13), 1495–1503. doi:10.1016/S0167-8655(02)00109-5

Ishibuchi, H., & Nakashima, T. (1999). Evolution of reference sets in nearest neighbor classification. In *Selected papers from the Second Asia-Pacific Conference on Simulated Evolution and Learning on Simulated Evolution and Learning* (LNCS 1585, pp. 82-89).

Ishibuchi, H., Nakashima, T., & Nii, M. (2001). Learning of neural networks with GA-based instance selection. In *Proceedings of the 20th North American Fuzzy Information Processing Society International Conference,* Vancouver, Canada (Vol. 4, pp. 2102-2107).

Kavsek, B., & Lavrac, N. (2006). APRIORI-SD: Adapting association rule learning to subgroup discovery. *Applied Artificial Intelligence*, *20*(7), 543–583. doi:10.1080/08839510600779688

Kennedy, J., Eberhart, R. C., & Shi, Y. (2001). *Swarm intelligence*. San Francisco: Morgan Kaufmann Publishers.

Kibbler, D., & Aha, D. W. (1987). Learning representative exemplars of concepts: An initial case of study. In *Proceedings of the 4th International Workshop on Machine Learning,* Irvine, CA (pp. 24-30).

Kim, K. (2006). Artificial neural networks with evolutionary instance selection for financial forecasting. *Expert Systems with Applications*, *30*, 519–526. doi:10.1016/j.eswa.2005.10.007

Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE*, *78*, 1464–1480. doi:10.1109/5.58325

Kubat, M., Holte, R. C., & Matwin, S. (1998). Machine learning for the detection of oil spills in satellite radar images. *Machine Learning*, *30*(2-3), 195–215. doi:10.1023/A:1007452223027

Kubat, M., & Matwin, S. (1997). Addressing the course of imbalanced training sets: One-sided selection. In *Proceedings of the 14th International Conference on Machine Learning,* Nashville, TN (pp. 179-186).

Kuncheva, L. I. (1995). Editing for the k-nearest neighbors rule by a genetic algorithm. *Pattern Recognition Letters*, *16*, 809–814. doi:10.1016/0167-8655(95)00047-K

Kuncheva, L. I., & Bezdek, J. C. (1998). Nearest prototype classification: Clustering, genetic algorithms, or random search? *IEEE Transactions on Systems, Man, and Cybernetics*, *28*(1), 160–164. doi:10.1109/5326.661099

Laurikkala, J. (2001). Improving identification of difficult small classes by balancing class distribution. In *Proceedings of the 8th Conference on Artificial Intelligence in Medicine in Europe,* Cascais, Portugal (pp. 63-66).

Lavrac, N., Kavsek, B., Flach, P., & Todorovski, L. (2004). Subgroup discovery with CN2-SD. *Journal of Machine Learning Research*, *5*, 153–188.

Liu, H., Hussain, F., Lim, C., & Dash, M. (2002). Discretization: An enabling technique. *Data Mining and Knowledge Discovery*, *6*(4), 393–423. doi:10.1023/A:1016304305535

Liu, H., & Motoda, H. (Eds.). (2001). *Instance selection and construction for data mining*. New York: Springer.

Liu, H., & Motoda, H. (2002). On issues of instance selection. *Data Mining and Knowledge Discovery*, *6*(2), 115–130. doi:10.1023/A:1014056429969

Liu, H., & Motoda, H. (Eds.). (2007). *Computational methods of feature selection*. New York: Chapman & Hall.

Lozano, M., Sotoca, J. M., Sánchez, J. S., Pla, F., Pekalska, E., & Duin, R. P. W. (2006). Experimental study on prototype optimisation algorithms for prototype-based classification in vector spaces. *Pattern Recognition*, *39*(10), 1827–1838. doi:10.1016/j.patcog.2006.04.005

Nanni, L., & Lumini, A. (2008). Particle swarm optimization for prototype reduction. *Neurocomputing*, *72*, 1092–1097. doi:10.1016/j.neucom.2008.03.008

Newman, D. J., Hettich, S., Blake, C. L., & Merz, C. J. (1998). *UCI repository of machine learning databases*. Irvine, CA: University of California, Irvine, Department of Information and Computer Sciences. Retrieved from http://www.ics.uci.edu/~mlearn/MLRepository.html

Oh, I., Lee, J., & Moon, B. (2004). Hybrid genetic algorithms for feature selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*(11), 1424–1437. doi:10.1109/TPAMI.2004.105

Ong, Y. S., Krasnogor, N., & Ishibuchi, H. (2007). Special issue on memetic algorithms. *IEEE Transactions on Systems, Man and Cybernetics . Part B*, *37*(1), 2–5.

Papadopoulos, A. N., & Manolopoulos, Y. (2004). *Nearest neighbor search: A database perspective*. Berlin, Germany: Springer-Verlag.

Paredes, R., & Vidal, E. (2006). Learning prototypes and distances: A prototype reduction technique based on nearest neighbor error minimization. *Pattern Recognition*, *39*(2), 180–188. doi:10.1016/j.patcog.2005.06.001

Provost, F. J., & Kolluri, V. (1999). A survey of methods for scaling up inductive learning algorithms. *Data Mining and Knowledge Discovery*, *2*, 131–169. doi:10.1023/A:1009876119989

Pyle, D. (1999). *Data preparation for data mining*. San Francisco: Morgan Kaufmann.

Quinlan, J. R. (1993). *C4.5: Programs for machine learning*. San Francisco: Morgan Kaufmann.

Riquelme, J. C., Aguilar, J. S., & Toro, M. (2003). Finding representative patterns with ordered projections. *Pattern Recognition*, *36*(4), 1009–1018. doi:10.1016/S0031-3203(02)00119-X

Ros, F., Guillaume, S., Pintore, M., & Chretien, J. R. (2008). Hybrid genetic algorithm for dual selection. *Pattern Analysis & Applications*, *11*, 179–198. doi:10.1007/s10044-007-0089-3

Sanchez, J. S., Barandela, R., Marques, A. I., Alejo, R., & Badenas, J. (2003). Analysis of new techniques to obtain quaylity training sets. *Pattern Recognition Letters*, *24*, 1015–1022. doi:10.1016/S0167-8655(02)00225-8

Sebban, M., Nock, R., Chauchat, J. H., & Rakotomalala, R. (2000). Impact of learning set quality and size on decision tree performances. *International Journal of Computers . Systems and Signals*, *1*(1), 85–105.

Shakhnarovich, G., Darrel, T., & Indyk, P. (Eds.). (2006). *Nearest-neighbor methods in learning and vision: Theory and practice*. Cambridge, MA: MIT Press.

Sierra, B., Lazkano, E., Inza, I., Merino, M., Larrañaga, P., & Quiroga, J. (2001). Prototype selection and feature subset selection by estimation of distribution algorithms. A case study in the survival of cirrhotic patients treated with TIPS. *Artificial Intelligence in Medicine* (LNAI 2101, pp. 20-29).

Tajbakhsh, A., Rahmati, M., & Mirzaei, A. (2009). Intrusion detection using fuzzy association rules. *Applied Soft Computing*, *9*(2), 462–469. doi:10.1016/j.asoc.2008.06.001

Wilson, D. R., & Martinez, T. R. (1997). Instance pruning techniques. In *Proceedings of the 14th International Conference on Machine Learning,* Nashville, TN (pp. 403-411).

Wilson, D. R., & Martinez, T. R. (2000). Reduction techniques for instance-based learning algorithms. *Machine Learning*, *38*, 257–286. doi:10.1023/A:1007626913721

Wu, S., & Olafsson, S. (2006). *Optimal instance selection for improved decision tree induction*. Paper presented at the 2006 IIE Annual Conference and Exhibition, Orlando, FL.

Yang, Q., & Wu, X. (2006). 10 challenging problems in data mining research. *International Journal of Information Technology and Decision Making*, *5*(4), 597–604. doi:10.1142/S0219622006002258

APPENDIX: ACRONYMS TABLE

In this appendix, a table with all the acronyms employed on the text is provided (Table 1). For each acronym, it is shown its meaning and the page where it was defined:

Table 1.

Acronym	Meaning	Page
EIS	Evolutionary Instance Selection	4
EIS-PS	Evolutionary Instance Selection – Prototype Selection	4
EIS-TSS	Evolutionary Instance Selection – Training Set Selection	4
EUS	Evolutionary Under Sampling	23
EPG	Evolutionary Prototype Generation	4
GGA	Generational Genetic Algorithm	10
GOCBR	Global Optimization of feature weighting and instance election using genetic algorithms for Case Based Reasoning	26
HGA	Hybrid Genetic Algorithm	26
IG	Instance Generation	3
IGA	Intelligent Genetic Algorithm	18
IS	Instance Selection	3
PBIL	Population Based Incremental Learning	10
PG	Prototype Generation	3
PS	Prototype Selection	3
PSO	Particle Swarm Optimization	28
SSGA	Steady State Genetic Algorithm	10
SSMA	Steady State Memetic Algorithm	18
TSS	Training Set Selection	3

Joaquín Derrac received the MSc degree in computer science from the University of Granada, Granada, Spain, in 2008. He is currently a PhD 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.

Salvador García received the MSc and PhD 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.

Francisco Herrera received the MSc degree in mathematics in 1988 and the PhD 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 Systesms, 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 the editorial board of the journals:* Fuzzy Sets and Systems, Applied Intelligence, Knowledge and Information Systems, Information Fusion, Evolutionary Intelligence, International Journal of Hybrid Intelligent Systems, Memetic Computation, International Journal of Computational Intelligence Research, The Open Cybernetics and Systemics Journal, Recent Patents on Computer Science, Journal of Advanced Research in Fuzzy and Uncertain Systems, International Journal of Information Technology and Intelligent and Computing, *and* Journal of Artificial Intelligence and Soft Computing Research. *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.*

2. Coevolutionary Algorithms for Enhancing Nearest Neighbor Classification

The journal papers associated to this part are:

2.1 IFS-CoCo: Instance and Feature Selection Based on Cooperative Coevolution With Nearest Neighbor Rule

- J. Derrac, S. García, F. Herrera, IFS-CoCo: Instance and Feature Selection Based on Cooperative Coevolution With Nearest Neighbor Rule. Pattern Recognition, 43 (2010) 20822105 doi: 10.1016/j.patcog.2009.12.012
  - Status: **Published**.
  - Impact Factor (JCR 2010): 2.682.
  - Subject Category: Computer Science, Artificial Intelligence. Ranking 15 / 108 (**Q1**).
  - Subject Category: Engineering, Electrical & Electronic. Ranking 18 / 247 (**Q1**).

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

A R T I C L E I N F O

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

A B S T R A C T

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.

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.

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.

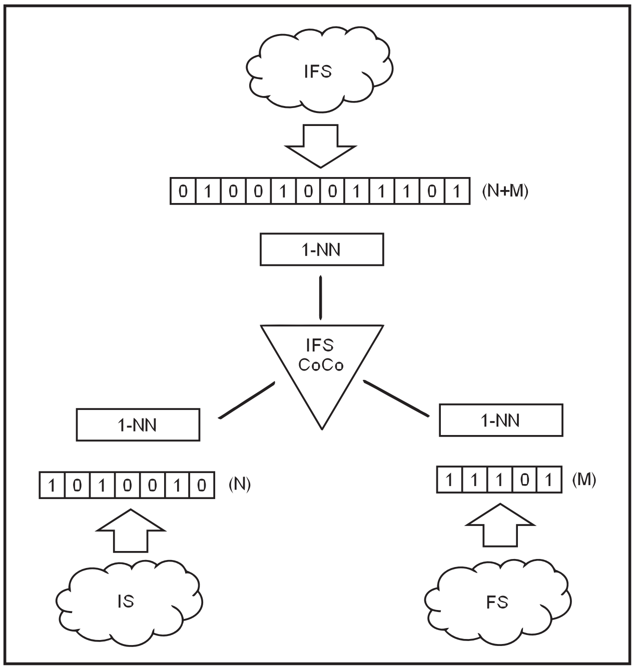


Fig. 1. Population scheme of IFS-CoCo.

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.

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

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.

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

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:

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

  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:

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

  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:

$$Fitness(J) = \alpha \cdot clasRate(J) + (1\text{-}\alpha) \cdot Reduction(J) \tag{5}$$

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:

$$Fitness(J) = \alpha \cdot \beta \cdot clasRate(J) + (1\text{-}\alpha) \cdot ReductionIS(J) + (1\text{-}\beta) \cdot ReductionFS(J) \tag{6}$$

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.

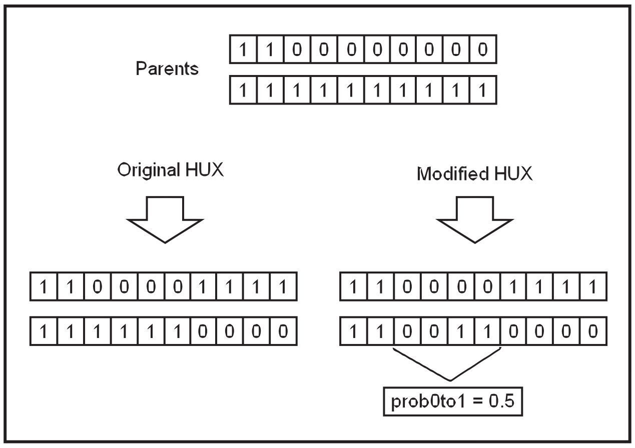


Fig. 2. The modified HUX crossing operator.

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.

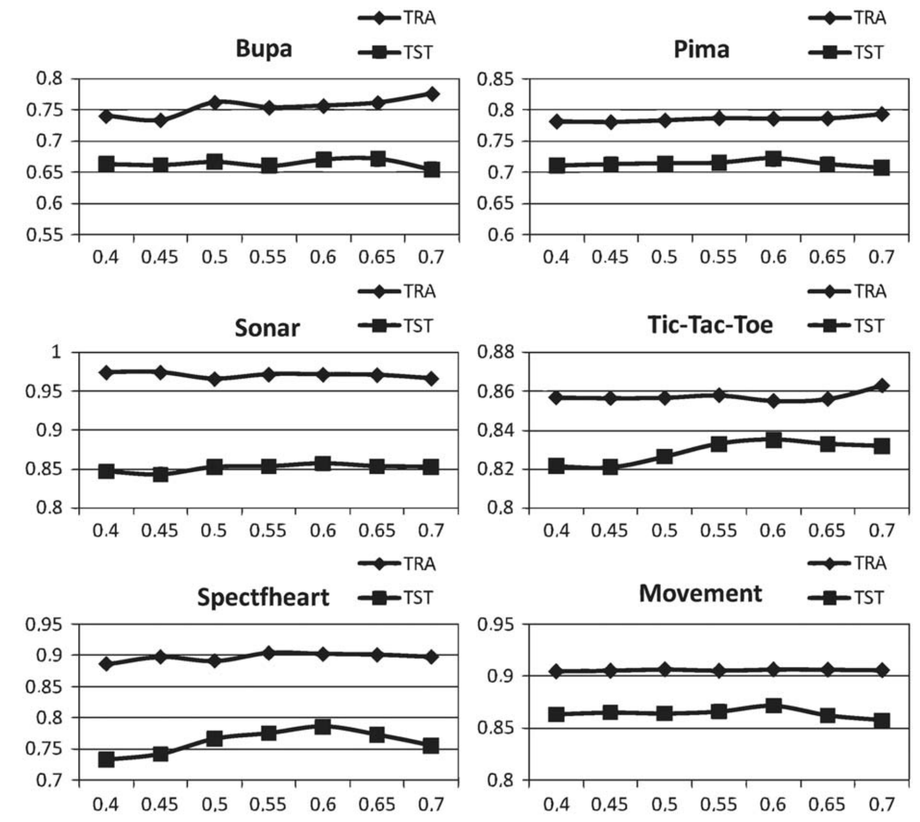


Fig. 3. Results in accuracy by employing different values (0.4–0.7) of the α parameter.

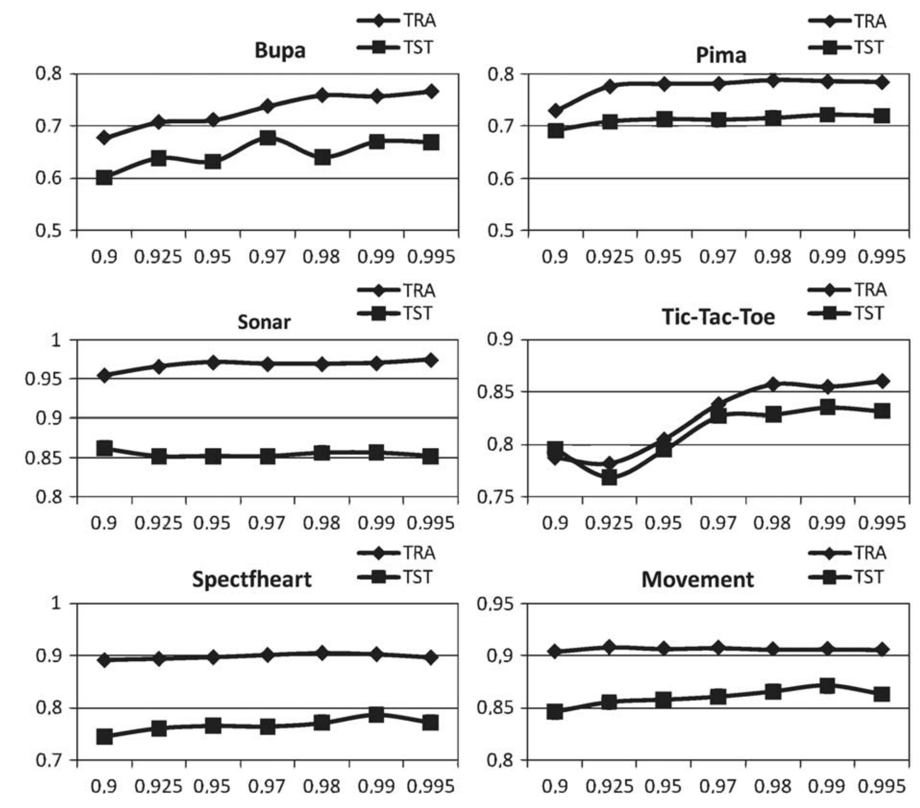


Fig. 4. Results in accuracy by employing different values (0.9–0.995) of the β parameter.

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

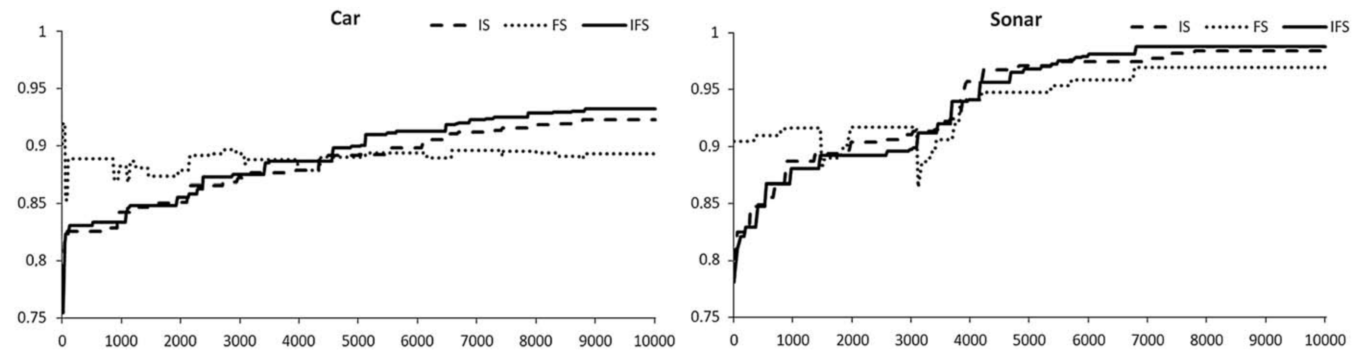


Fig. 5. Map of convergence of IFS-CoCo on Car and Sonar data sets.

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	**96.56 ± 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	**86.66 ± 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	**91.29 ± 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	**93.69 ± 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	**86.83 ± 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	**99.05 ± 1.47**
Avg.	93.22 ± 0.73	**92.23 ± 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	**96.56 ± 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	**86.66 ± 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	**91.29 ± 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	**93.69 ± 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	**86.83 ± 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	**99.05 ± 1.47**
Avg.	93.22 ± 0.73	**92.23 ± 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	**96.56 ± 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	**86.66 ± 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	**91.29 ± 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	**93.69 ± 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	**86.83 ± 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	**99.05 ± 1.47**
Avg.	93.22 ± 0.73	**92.23 ± 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	**93.09 ± 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	**79.16 ± 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	**89.26 ± 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	**86.68 ± 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	**79.18 ± 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	**98.80 ± 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	**87.69 ± 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	**93.09 ± 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	**79.16 ± 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	**89.26 ± 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	**86.68 ± 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	**79.18 ± 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	**98.83 ± 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	**87.69 ± 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	**93.09 ± 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	**79.16 ± 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	**89.26 ± 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	**86.68 ± 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	**79.18 ± 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	**98.80 ± 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	**87.69 ± 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.

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

- J. Derrac, I. Triguero, S. García, F. Herrera, IFS-CoCo: Integrating Instance Selection, Instance Weighting, and Feature Weighting for Nearest Neighbor Classifiers by Coevolutionary Algorithms. IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics 42:5 (2012) 13831397 doi: 10.1109/TSMCB.2012.2191953
  - Status: **Published**.
  - Impact Factor (JCR 2011): 3.080.
  - Subject Category: Automation & Control Systems. Ranking 2 / 58 (**Q1**).
  - Subject Category: Computer Science, Artificial Intelligence. Ranking 10 / 111 (**Q1**).
  - Subject Category: Computer Science, Cybernetics. Ranking 1 / 20 (**Q1**).

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

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

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

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

I. Introduction

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

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

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

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

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

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

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

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

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

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

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

Digital Object Identifier 10.1109/TSMCB.2012.2191953

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

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

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

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

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

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

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

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

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

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

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

A. Coevolution: Main Trends and Key Issues

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

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

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

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

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

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

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

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

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

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

B. IS

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

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

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

C. Weighting Schemes

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

III. CIW-NN

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

A. Population Scheme

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

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

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

TABLE I
CIW-NN POPULATION'S CHARACTERISTICS

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

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

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

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

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

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

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

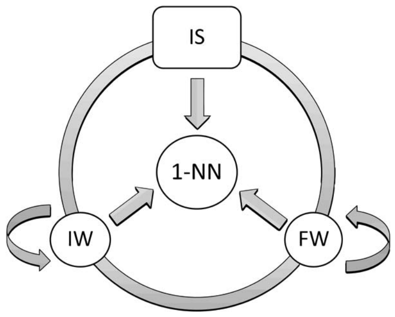


Fig. 1. Evolutionary cycle of CIW-NN. IS, FW, and IW populations evolve simultaneously, spending an epoch in turns. IS epochs only spend one generation, while FW and IW epochs spend several. Individuals of every population cooperate through the fitness function (a 1-NN classifier).

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

Fig. 2. Coevolutionary model.

B. Coevolutionary Model

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

CIW-NN only selects one individual per population (the best) as a collaborator. Although other schemes, such as selecting a set of collaborators per population, could be defined, the employment of just one collaborator has a unique advantage: The evaluation of any new individual only consumes one evaluation of the fitness function. By contrast, employing any other scheme would lead to a quadratic increase in the number of evaluations needed to characterize a new individual (if Z collaborators per each of the three populations are selected, a new individual will require Z^2 evaluations to consider all the possible combinations). Given the nature of the problems tackled by CIW-NN, which are characterized by a costly fitness function, computationally speaking (more costly, by far, than, for example, the ones usually considered in function optimization problems; see Section III-C), this decrease in the evaluation requirements is indispensable.

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

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

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

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

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

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

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

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

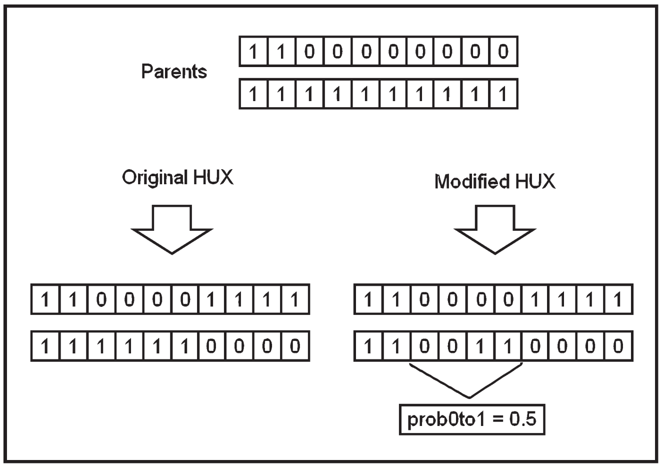


Fig. 3. HUX crossover operator exchanges exactly half of the nonmatching alleles, selected randomly. In our modified version of HUX, an allele valued with 1 has a probability $prob0to1$ of maintaining its value when it is selected to be exchanged.

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

The SSGA has the following features.

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

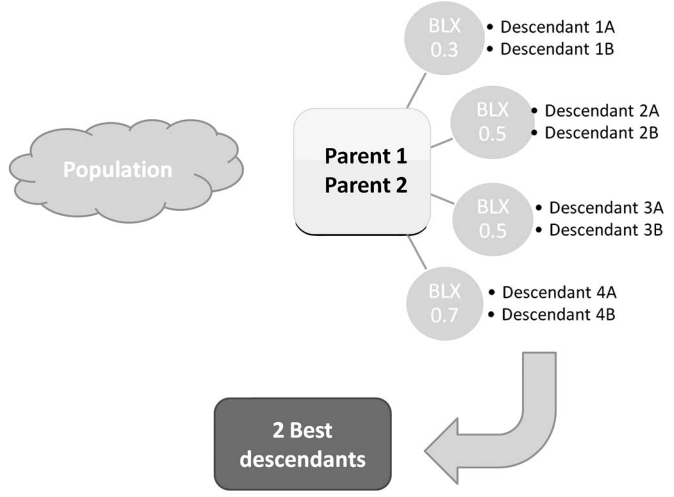


Fig. 4. Scheme of application of the 2BLX0.3–4BLX0.5–2BLX0.7 multiple-descendant crossover operator.

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

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

C. Cooperation in CIW-NN: The Fitness Function

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

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

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

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

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

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

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

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

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

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

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

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

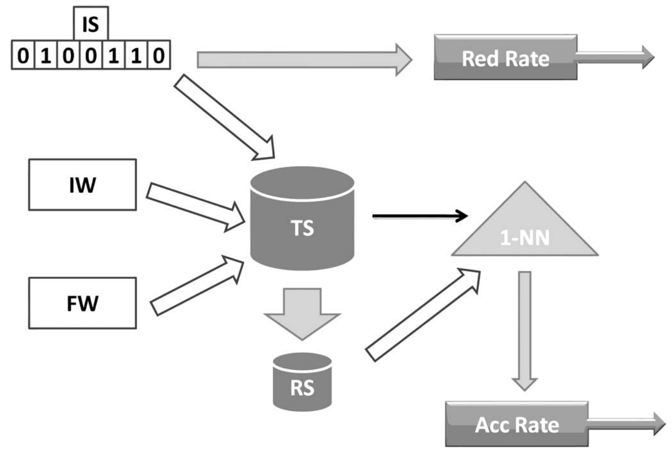


Fig. 5. Example fitness assignment for an IS individual. $RedRate$ is computed directly. To compute $AccRate$, IS, FW, and IW chromosomes are applied to the training set TS, obtaining the reference set RS, which is employed by the 1-NN classifier to estimate accuracy.

D. Fitness Assignment

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

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

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

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

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

$$Fitness(K) = AccRate(K). \tag{7}$$

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

TABLE II
SUMMARY DESCRIPTION OF SMALL DATA SETS

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

TABLE III
SUMMARY DESCRIPTION OF LARGE DATA SETS

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

IV. EXPERIMENTAL FRAMEWORK

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

A. Classification Problems

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

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

B. Comparison Methods

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

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

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

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

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

TABLE IV
PARAMETER SPECIFICATION FOR THE METHODS OF THE STUDY

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

C. Performance Measures

We have selected the following performance measures.

1) **Accuracy**: It is defined as the number of successful hits relative to the total number of classifications. It has been by far the most commonly used metric for assessing the performance of classifiers for years [1], [4].
2) **Kappa**: It is an alternative to the accuracy rate, a method, known for decades, that compensates for random hits [55] in the same way as the Area Under the ROC curve measure. Cohen's kappa measure can be obtained using the expression

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

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

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

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

D. Statistical Tools for Analysis

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

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

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

V. RESULTS AND ANALYSIS

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

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

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

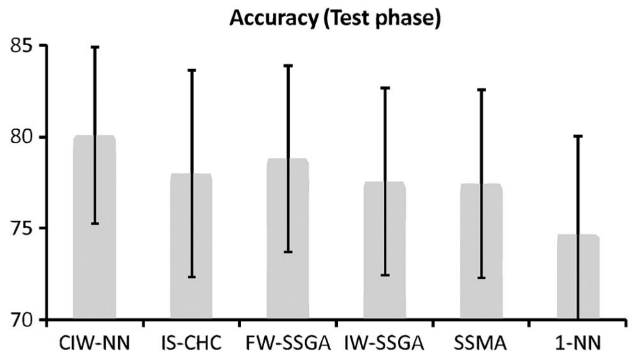


Fig. 6. Graphical comparison of accuracy in the test phase between CIW-NN and evolutionary proposals for k-NN-based classification.

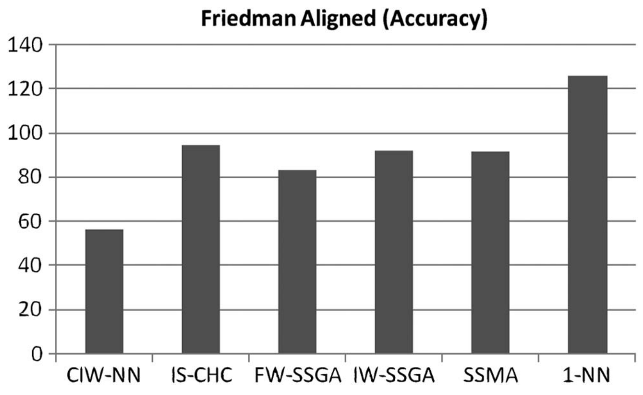


Fig. 7. Rankings computed by Friedman aligned-ranks procedures by using the accuracy measure.

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

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

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

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

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

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

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

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

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

Using these results, we can conclude the following.

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

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

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

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

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

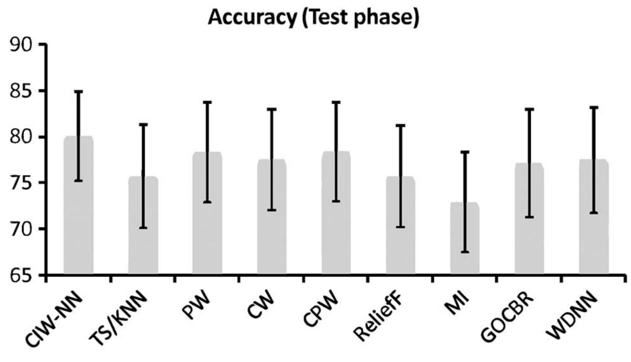


Fig. 8. Graphical comparison of accuracy in the test phase for CIW-NN and weighting methods.

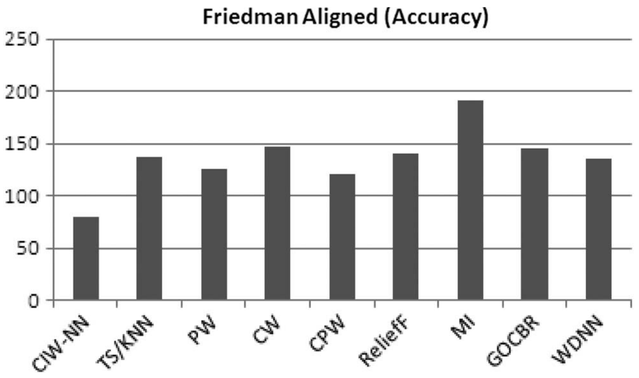


Fig. 9. Rankings computed by Friedman aligned-ranks procedure by using the accuracy measure.

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

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

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

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

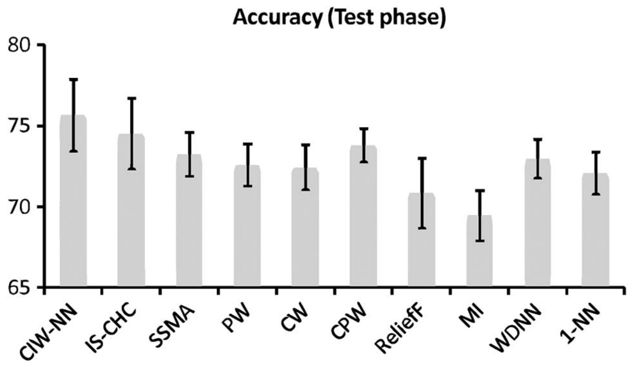


Fig. 10. Graphical comparison of accuracy in test phase in large domains.

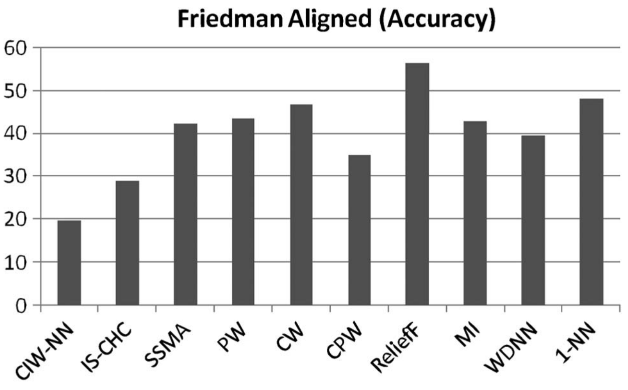


Fig. 11. Rankings of Friedman aligned-ranks test using accuracy in large domains.

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

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

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

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

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

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

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

With these results, we can conclude the following.

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

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

D. Convergence Analysis

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

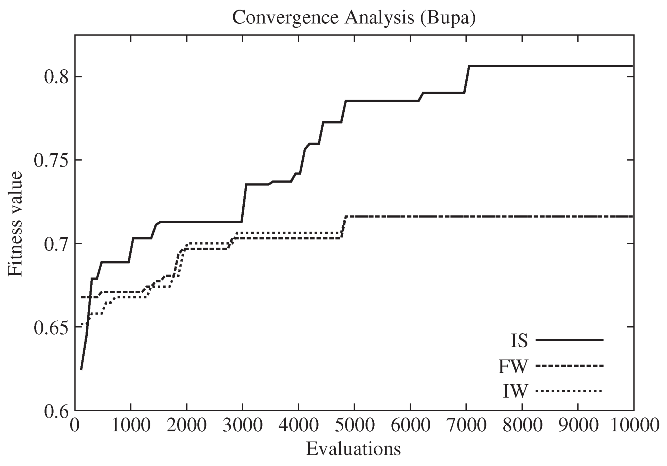


Fig. 12. Convergence analysis for CIW-NN in the Bupa problem. Critical points are found at 3000 and 4800 evaluations approximately, where the progress of a population allows the rest to escape from local maximums.

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

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

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

VI. Conclusion

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

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

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

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

Acknowledgment

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

References

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

[21] G. L. Pappa and A. A. Freitas, *Automating the Design of Data Mining Algorithms: An Evolutionary Computation Approach*. New York: Springer-Verlag, ser. Natural Computing, 2009.

[22] A. Fernandez, S. Garcia, J. Luengo, E. Bernado-Mansilla, and F. Herrera, “Genetics-based machine learning for rule induction: State of the art, taxonomy and comparative study,” *IEEE Trans. Evol. Comput.*, vol. 14, no. 6, pp. 913–941, Dec. 2010.

[23] J. R. Cano, F. Herrera, and M. Lozano, “Using evolutionary algorithms as instance selection for data reduction in KDD: An experimental study,” *IEEE Trans. Evol. Comput.*, vol. 7, no. 6, pp. 561–575, Dec. 2003.

[24] L. I. Kuncheva, “Editing for the k-nearest neighbors rule by a genetic algorithm,” *Pattern Recognit. Lett.*, vol. 16, no. 6, pp. 809–814, Aug. 1995.

[25] S.-Y. Ho, C.-C. Liu, and S. Liu, “Design of an optimal nearest neighbor classifier using an intelligent genetic algorithm,” *Pattern Recognit. Lett.*, vol. 23, no. 13, pp. 1495–1503, Nov. 2002.

[26] S. García, J. R. Cano, and F. Herrera, “A memetic algorithm for evolutionary prototype selection: A scaling up approach,” *Pattern Recognit.*, vol. 41, no. 8, pp. 2693–2709, Aug. 2008.

[27] S. García and F. Herrera, “Evolutionary undersampling for classification with imbalanced datasets: Proposals and taxonomy,” *Evol. Comput.*, vol. 17, no. 3, pp. 275–306, Nov. 2009.

[28] A. Cervantes, I. Galván, and P. Isasi, “AMPSO: A new particle swarm method for nearest neighborhood classification,” *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 5, pp. 1082–1091, Oct. 2009.

[29] M. A. Potter and K. A. D. Jong, “Cooperative coevolution: An architecture for evolving coadapted subcomponents,” *Evol. Comput.*, vol. 8, no. 1, pp. 1–29, 2000.

[30] M. Li and Z. Wang, “A hybrid coevolutionary algorithm for designing fuzzy classifiers,” *Inf. Sci.*, vol. 179, no. 12, pp. 1970–1983, May 2009.

[31] P. Gançarski, A. Blansché, and A. Wania, “Comparison between two coevolutionary feature weighting algorithms in clustering,” *Pattern Recognit.*, vol. 41, no. 3, pp. 983–994, Mar. 2008.

[32] Z. Yang, K. Tang, and X. Yao, “Large scale evolutionary optimization using cooperative coevolution,” *Inf. Sci.*, vol. 178, no. 15, pp. 2985–2999, Aug. 2008.

[33] Y. Wang and A. Nakao, “On cooperative and efficient overlay network evolution based on a group selection pattern,” *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 40, no. 3, pp. 656–667, Jun. 2010.

[34] N. García-Pedrajas, C. Hervás-Martínez, and D. Ortiz-Boyer, “Cooperative coevolution of artificial neural network ensembles for pattern classification,” *IEEE Trans. Evol. Comput.*, vol. 9, no. 3, pp. 271–302, Jun. 2005.

[35] N. García-Pedrajas, J. A. R. del Castillo, and D. Ortiz-Boyer, “A cooperative coevolutionary algorithm for instance selection for instance-based learning,” *Mach. Learn.*, vol. 78, no. 3, pp. 381–420, Mar. 2010.

[36] J. Derrac, S. García, and F. Herrera, “IFS-CoCo: Instance and feature selection based on cooperative coevolution with nearest neighbor rule,” *Pattern Recognit.* , vol. 43, no. 6, pp. 2082–2105, Jun. 2010.

[37] A. M. Sánchez, M. Lozano, P. Villar, and F. Herrera, “Hybrid crossover operators with multiple descendents for real-coded genetic algorithms: Combining neighborhood-based crossover operators,” *Int. J. Intell. Syst.*, vol. 24, no. 5, pp. 540–567, May 2009.

[38] C. Rosin and R. Belew, “New methods for competitive coevolution,” *Evol. Comput.*, vol. 15, no. 1, pp. 1–29, 1997.

[39] C.-K. Goh and K. C. Tan, “A competitive–cooperative coevolutionary paradigm for dynamic multiobjective optimization,” *IEEE Trans. Evol. Comput.*, vol. 13, no. 1, pp. 103–127, Feb. 2009.

[40] R. P. Wiegand and J. Sarma, “Spatial embedding and loss of gradient in cooperative coevolutionary algorithms,” in *Proc. 8th Int. Conf.—PPSN*, Birmingham, U.K., Sep. 18–22, 2004, vol. 3242, pp. 912–921.

[41] X. Li and X. Yao, “Tackling high dimensional nonseparable optimization problems by cooperatively coevolving particle swarms,” in *Proc. IEEE CEC*, 2009, vol. 9, pp. 1546–1553.

[42] E. Popovici and K. A. D. Jong, “Sequential versus parallel cooperative coevolutionary algorithms for optimization,” in *Proc. IEEE CEC*, Vancouver, BC, Canada, 2006, vol. 3242, pp. 1610–1617.

[43] E. Popovici and K. A. D. Jong, “The effects of interaction frequency on the optimization performance of cooperative coevolution,” in *Proc. GECCO*, Seattle, WA, Jul. 8–12, 2006, pp. 353–360.

[44] L. Panait, S. Luke, and J. F. Harrison, “Archive-based cooperative coevolutionary algorithms,” in *Proc. GECCO*, Seattle, WA, Jul. 8–12, 2006, pp. 345–352.

[45] S. García, J. Derrac, J. R. Cano, and F. Herrera, “Prototype selection for nearest neighbor classification: Taxonomy and empirical study,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 3, pp. 417–435, Mar. 2012.

[46] F. Fernández and P. Isasi, “Local feature weighting in nearest prototype classification,” *IEEE Trans. Neural Netw.*, vol. 19, no. 1, pp. 40–53, Jan. 2008.

[47] I. Kononenko, “Estimating attributes: Analysis and extensions of RELIEF,” in *Proc. Eur. Conf. Mach. Learn.*, Catania, Italy, 1994, pp. 171–182.

[48] R. Paredes and E. Vidal, “Learning weighted metrics to minimize nearest-neighbor classification error,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 7, pp. 1100–1110, Jul. 2006.

[49] L. J. Eshelman, “The CHC adaptative search algorithm: How to have safe search when engaging in nontraditional genetic recombination,” in *Foundations of Genetic Algorithms*, G. J. E. Rawlins, Ed. San Mateo, CA: Morgan Kaufmann, 1991, pp. 265–283.

[50] J. D. Kelly and L. Davis, “A hybrid genetic algorithm for classification,” in *Proc. 12th IJCAI*, Sydney, Australia, 1991, vol. 2, pp. 645–650.

[51] H. Ahn and K. Kim, “Bankruptcy prediction modeling with hybrid case-based reasoning and genetic algorithms approach,” *Appl. Soft Comput.*, vol. 9, no. 2, pp. 599–607, Mar. 2009.

[52] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs (2nd, Ext. ed.)*. New York: Springer-Verlag, 1994.

[53] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, and F. Herrera, “Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework,” *J. Multiple-Valued Logic Soft Comput.*, vol. 17, no. 2/3, pp. 255–287, 2011.

[54] M. A. Tahir, A. Bouridane, and F. Kurugollu, “Simultaneous feature selection and feature weighting using hybrid tabu search/ k-nearest neighbor classifier,” *Pattern Recognit. Lett.*, vol. 28, no. 4, pp. 438–446, Mar. 2007.

[55] J. Cohen, “A coefficient of agreement for nominal scales,” *Edu. Psychol. Meas.*, vol. 20, no. 1, pp. 37–46, Apr. 1960.

[56] S. García, A. Fernández, J. Luengo, and F. Herrera, “Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power,” *Inf. Sci.*, vol. 180, no. 10, pp. 2044–2064, May 2010.

[57] L. Panait, “Theoretical convergence guarantees for cooperative coevolutionary algorithms,” *Evol. Comput.*, vol. 18, no. 4, pp. 581–615, Nov. 2010.

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

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

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

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

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

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

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

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

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

3. Fuzzy Rough Sets and Evolutionary Algorithms in Nearest Neighbor Classification

The journal papers associated to this part are:

3.1 Enhancing Evolutionary Instance Selection Algorithms by Means of Fuzzy Rough Set Based Feature Selection

- J. Derrac, C. Cornelis, S. García, F. Herrera, Enhancing Evolutionary Instance Selection Algorithms by Means of Fuzzy Rough Set Based Feature Selection. Information Sciences, 186 (2012) 7392 doi: 10.1016/j.ins.2011.09.027
  - Status: **Published**.
  - Impact Factor (JCR 2011): 2.833.
  - Subject Category: Computer Science, Information Systems. Ranking 9 / 135 (**Q1**).

Information Sciences 186 (2012) 73–92

Contents lists available at SciVerse ScienceDirect

Information Sciences

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

Enhancing evolutionary instance selection algorithms by means of fuzzy rough set based feature selection

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

[a] Dept. of Computer Science and Artificial Intelligence, CITIC-UGR, Research Center on Information and Communications Technology, University of Granada, 18071 Granada, Spain
[b] Dept. of Applied Mathematics and Computer Science, Ghent University, Gent, Belgium
[c] Dept. of Computer Science, University of Jaén, 23071 Jaén, Spain

A R T I C L E   I N F O

Article history:
Received 9 October 2010
Received in revised form 10 June 2011
Accepted 18 September 2011
Available online 29 September 2011

Keywords:
Instance selection
Feature selection
Rough sets
Evolutionary algorithms
Nearest neighbor

A B S T R A C T

In recent years, fuzzy rough set theory has emerged as a suitable tool for performing feature selection. Fuzzy rough feature selection enables us to analyze the discernibility of the attributes, highlighting the most attractive features in the construction of classifiers. However, its results can be enhanced even more if other data reduction techniques, such as instance selection, are considered.

In this work, a hybrid evolutionary algorithm for data reduction, using both instance and feature selection, is presented. A global process of instance selection, carried out by a steady-state genetic algorithm, is combined with a fuzzy rough set based feature selection process, which searches for the most interesting features to enhance both the evolutionary search process and the final preprocessed data set. The experimental study, the results of which have been contrasted through nonparametric statistical tests, shows that our proposal obtains high reduction rates on training sets which greatly enhance the behavior of the nearest neighbor classifier.

1. Introduction

Classification is one of the best known tasks in machine learning [5,72]. Starting from an already processed training set, machine learning methods are able to extract knowledge from the data, which can be used to characterize new samples and classify them into classes already specified by the domain of the problem.

In recent years, there has been a manifold increase in the size of the data which these machine learning methods must manage [6]. Researchers in many application fields have developed more efficient and accurate data acquisition methods, which have allowed them to face greater and more difficult problems than before [45]. Therefore, the amount of data extracted to analyze those new challenges has grown to a point at which many classical data mining methods do not work properly, or, at least, suffer several drawbacks in their application.

Data reduction [54] is a data preprocessing task which can be applied to ease the problem of dealing with large amounts of data. Its main objective is to reduce the original data by selecting the most representative information. In this way, it is possible to avoid excessive storage and time complexity, improving the results obtained by any data mining application. The best known data reduction processes are feature selection (FS) [43], feature generation [28], attribute discretization [39], instance generation [65,66] and instance selection (IS) [21,41,42].

* Corresponding author. Tel.: +34 958 240598; fax: +34 958 243317.
E-mail addresses: jderrac@decsai.ugr.es (J. Derrac), Chris.Cornelis@UGent.be (C. Cornelis), sglopez@ujaen.es (S. García), herrera@decsai.ugr.es (F. Herrera).

The K-Nearest Neighbors classifier (K-NN) [13,48,58] can be greatly enhanced when using these data reduction techniques. It is a nonparametric classifier which simply uses the entire input data set to establish the classification rule. Thus, the effectiveness of the classification process performed by the K-NN classifier relies mainly on the quality of the training data. Also, it is important to note that its main drawback is its relative inefficiency as the size of the problem increases, regarding both the number of examples in the data set and the number of attributes which will be used in the computation of the similarity function (distance) [11,33,68]. The K-NN classifier is one of the most relevant algorithms in data mining [73], being the best known Lazy Learning [1] method.

Recently, rough set theory (RST) [50,53] has been used to tackle several data mining problems with success. Focusing on the FS problem, RST can be used as a tool to extract a minimal set of features from the original data set (*decision reducts*), preserving the underlying semantics of the data while allowing reasonable generalization capabilities for the classifier [10,74]. This approach can be enhanced in several ways; for example, tolerance-based rough sets [60] provide an advanced way of defining approximation spaces and related similarity measures [46,61,62].

In addition, fuzzy logic [75] can also be hybridized with RST, obtaining as a result fuzzy rough feature selection methods, which offer greater flexibility and better potential to produce good-sized, high-quality feature subsets than the crisp ones [12,30,34,35,67]. Another key trait of fuzzy rough feature selection methods is that they can be applied directly over data sets representing continuous data, in contrast with pure RST feature reducers, which cannot be applied over continuous data sets without discretizing them at a previous step. Although discretization has proven to be a good technique for solving this kind of issues [39], in the K-NN classifier the use of continuous values is preferred due to the intrinsic characteristics of its decision rule (in fact, much research has been carried out in the opposite direction, turning discrete-based similarities values for the K-NN classifier into continuous ones [70]).

Evolutionary algorithms (EAs) [17] are general-purpose search algorithms that use principles inspired by nature to evolve solutions to problems. They have been successfully applied in different data mining problems [19,25,49]. Given that IS and FS tasks can be defined as combinatorial problems, it is possible to carry them out by using EAs [15]. In fact, many successful evolutionary proposals (most of them based on Genetic Algorithms (GAs)) have been developed to tackle them [2,7,20,29,31,32,38,47]. In particular, our previous work [14] proposes an evolutionary method for dealing with both IS and FS tasks simultaneously, using a ensemble based on preprocessed training sets.

In this work we present a new hybrid approach considering both fuzzy RST based FS and evolutionary IS, which we denote as EIS-RFS (evolutionary instance selection enhanced by Rough set based feature selection). A steady-state GA is developed to conduct the search for a suitable subset of instances, whereas the most useful features are selected by an heuristic-based fuzzy RST method. In this way, proper feature subsets can be selected during the search, taking advantage of using the information about indiscernibility already present in the training set.

Moreover, these subsets are considered within the framework of the GA, thus modifying the environment in which the instances are chosen. At the end of its application, EIS-RFS reports the best subsets found, which can be used to construct a reduced version of the training set, well suited to be used as a reference set for the K-NN classifier.

The performance of EIS-RFS is studied, comparing it with the K-NN classifier over unreduced data, considering different numbers of neighbors. Moreover, we test our approach further introducing a comparative study with several related techniques for IS and FS, considering a large set of standard classification problems. Finally, we also test its performance over large data sets, with a higher number of instances and features. All the results obtained have been contrasted using nonparametric statistical techniques [14], reinforcing the conclusions obtained.

The rest of the paper is organized as follows. In Section 2, some background about evolutionary IS and fuzzy RST based FS is given. In Section 3, the main characteristics of EIS-RFS are explained. Section 4 presents the experimental framework. Section 5 shows the achieved results. Section 6 summarizes our conclusions. Finally, Appendix A extends the description of the contrast estimation, one of the nonparametric tests employed in the study.

2. Background

This section covers the background information necessary to define and describe our proposal. It focuses on two topics: IS and FS as data reduction techniques (Section 2.1), and the use of fuzzy RST for FS (Section 2.2).

2.1. Instance and feature 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 [41]. 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 $(\mathcal{X}, \mathcal{A})$ be an information system, where $\mathcal{X} = \{x_1, \ldots, x_n\}$ and $\mathcal{A} = \{a_1, \ldots, a_m\}$ are finite, non-empty sets of instances and features. Then, let us assume that there is a training set TR which consists of N instances and M features ($M = |\mathcal{A}|$), and a test set TS composed of T instances ($TR \cup TS = (\mathcal{X}, \mathcal{A})$). Let $S \subseteq TR$ be the subset of selected samples that resulted from the execution of an 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 [21] 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 (for example, 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, subgroup discovery, neural networks [8,9,37]). In this work, we will focus on PS, since the nearest neighbor Rule will be used as the baseline rule to perform the classification process.

A key property of PS methods is given in [71]. There, Wilson and Martinez suggest that the determination of the k value in the K-NN classifier may depend on the proposal of the IS algorithm. Setting k as greater than 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 against noisy instances. Therefore, they state 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. In this study, we will test the differences found when using several values for k in EIS-RFS, although we recommend using the value $k = 1$, given that our EA needs to have the greatest possible sensitivity to noise during the reduction process. In this way, an evolutionary IS algorithm can better detect the noisy and redundant instances in order to find a subset of instances adapted to the simplest method of nearest neighbors.

Many approaches to PS have been developed [21]. Concerning evolutionary IS [15], the first contribution was made by Kuncheva et al. [38]. Interest in this field was increased by the study performed by Cano et al. [7], where a complete study of the use of EAs in IS was made. They concluded that EAs outperform classical algorithms both in reduction rates and classification accuracy. Therefore, research in this field has grown recently, with a wide number of noteworthy proposals [2,14,20,24,26,29].

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 [43].

Starting from the definition given for IS, FS can be defined as follows: Let us assume $\mathcal{A}, \mathcal{X}, TR$ and TS have already been defined. Let $B \subseteq \mathcal{A}$ be the subset of selected features that resulted from the execution of an FS algorithm over TR, then we classify a new pattern from TS by a data mining algorithm acting over TR, employing as a reference only the features selected in B.

There are three main categories into 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 [55].
- *Filtering methods*, where the selection criterion is independent of the learning algorithm (separability measures are employed to guide the selection) [27].
- *Embedded methods*, where the search for an optimal subset is built into the classifier construction [57].

As with IS methods, a large number of FS methods have been developed recently. Two of the most well known classical algorithms are forward sequential and backward sequential selection [40], which begin with a feature subset and sequentially add or remove features until the finalization of the algorithm. Some complete surveys, analyzing both classical and advanced approaches to FS, can be found in the literature [27,57,44].

2.2. Fuzzy RST for FS

In rough set analysis [51,52], each attribute a in $\mathcal{A}$ corresponds to an $\mathcal{X} \to V_a$ mapping, in which V_a is the value of a over $\mathcal{X}$. For every subset B of $\mathcal{A}$, the B-indiscernibility relation R_B is

$$R_B = \{(x,y) \in \mathcal{X}^2 \quad \text{and} \quad (\forall a \in B)(a(x) = a(y))\} \tag{1}$$

Therefore, R_B is an equivalence relation. Its equivalence classes $[x]_{R_B}$ can be used to approximate concepts, that is, subsets of the universe $\mathcal{X}$. Given $A \subseteq \mathcal{X}$, its lower and upper approximation with respect to R_B are defined by

$$R_B \downarrow A = \{x \in \mathcal{X} | [x]_{R_B} \subseteq A\} \tag{2}$$

$$R_B \uparrow A = \{x \in \mathcal{X} | [x]_{R_B} \cap A \neq \emptyset\} \tag{3}$$

A *decision system* $(\mathcal{X}, \mathcal{A} \cup \{d\})$ is a special kind of information system, used in the context of classification, in which $d (d \notin \mathcal{A})$ is a designated attribute called the decision attribute. Its equivalence classes $[x]_{R_d}$ are called decision classes. Given $B \subseteq A$, the B-positive region POS_B contains those objects from X for which the values of B allow to predict the decision class unequivocally:

$$POS_B = \bigcup_{x \in X} R_B \downarrow [x]_{R_d} \tag{4}$$

Indeed, if $x \in POS_B$, it means that whenever an instance has the same values as x for the attributes in B, it will also belong to the same decision class as x. The predictive ability with respect to d of the attributes in B is then measured by the following value (degree of dependency of d on B):

$$\gamma_B = \frac{|POS_B|}{|\mathcal{X}|} \tag{5}$$

A subset B of $\mathcal{A}$ is called a decision reduct if it satisfies $POS_B = POS_{\mathcal{A}}$, that is, B preserves the decision making power of $\mathcal{A}$, and moreover it cannot be further reduced; in other words, there is no proper subset B' of B so that $POS_{B'} = POS_{\mathcal{A}}$. If the latter constraint is lifted – B is not necessarily minimal – we call B a decision superreduct.

Instead of using a crisp equivalence relation R to represent objects' indiscernibility, we can also measure their approximate equality by means of a fuzzy relation R. Typically, we assume that R is at least a fuzzy tolerance relation; in other words, R is reflexive and symmetric.

Assuming that for a qualitative attribute a, the classical way of discerning objects is used, that is, $R_a(x,y) = 1$ if $a(x) = a(y)$ and $R_a(x,y) = 0$ otherwise, we can define, for any subset B of $\mathcal{A}$, the fuzzy B-indiscernibility relation by

$$R_B(x,y) = \mathcal{T}(\underbrace{R_a(x,y)}_{a\in B}) \tag{6}$$

in which $\mathcal{T}$ represents a t-norm. It can be seen that if only qualitative attributes (possibly originating from discretization) are used, then the traditional concept of B-indiscernibility relation is recovered.

For the lower approximation of a fuzzy set A in X by means of a fuzzy tolerance relation R, we adopt the definitions of [56]: given an implicator $\mathcal{I}$ and a t-norm $\mathcal{T}$, formulas (2) and (3) were paraphrased to define $R{\downarrow}A$ and $R{\uparrow}A$ by

$$(R \downarrow A)(y) = \inf_{x\in X} \mathcal{I}(R(x,y), \quad A(x)), \quad \forall y \in X \tag{7}$$

$$(R \uparrow A)(y) = \sup_{x\in X} \mathcal{T}(R(x,y), \quad A(x)), \quad \forall y \in X \tag{8}$$

Using fuzzy B-indiscernibility relations, we can define the fuzzy B-positive region by, for y in U,

$$POS_B(y) = \left(\bigcup_{x\in X} R_B \downarrow [\mathcal{X}_{R_d}]\right)(y) \tag{9}$$

This means that the fuzzy positive region is a fuzzy set in X, to which an object y belongs to the extent that its R_B-foreset is included into at least one of the decision classes.

While formula (9) provides the most faithful way to define the fuzzy positive region, it was shown in [12] that

$$POS_B(y) = (R_B \downarrow R_d y)(y) \tag{10}$$

becomes Eq. (9) when the decision feature is crisp.

Once we have fixed the fuzzy positive region, we can define an increasing [0, 1]-valued measure to gauge the degree of dependency of a subset of features on another subset of features. For FS it is useful to phrase this in terms of the dependency of the decision feature on a subset of the conditional features:

$$\gamma_B = \frac{|POS_B|}{|POS_{\mathcal{A}}|} \tag{11}$$

3. An evolutionary fuzzy RST based model for feature and instance selection: EIS-RFS

This section is devoted to analyzing and describing EIS-RFS, and its main components, from a bottom-up perspective. Therefore, the first step will be to describe the GA employed to conduct the search of the subsets of instances (Section 3.1). These subsets will be optimized with the inclusion of a fuzzy RST-based FS procedure (Section 3.2). Finally, the EIS-RFS framework will be defined as a cooperation between the former procedures (Section 3.3).

3.1. A steady-state GA for IS

The IS component of EIS-RFS is guided by an evolutionary method. Specifically, we have opted to develop a steady-state GA to accomplish this task.

Steady-state GAs are GAs in which only a reduced (and fixed) set of offspring are produced in each generation (usually one or two).Parents are selected to produce offspring and then a decision is made as to which individuals in the population will be selected for deletion in order to make room for the new offspring.

In the development of the steady-state GA for our approach, the following choices have been made:

- *Codification*: The steady-state GA will use binary chromosomes to represent the solutions. Each bit will represent the state of each instance in the training set (**1** if the instance is selected; **0** if it is deleted), as is usually done in Evolutionary IS approaches [15].

- *Selection of parents*: Binary tournament procedure will be used to select parents in each generation.
- *Crossover operator*: A two-point crossover operator has been considered. In each generation, this operator is applied twice, obtaining as a result two offspring.
- *Mutation operator*: The bit-flip mutation operator (changing the value of the selected allele from 0 to 1, and vice versa) is applied to each offspring produced, with a given probability per bit.
- *Replacement strategy*: The two worst individuals of the population are chosen for replacement, only if their fitness value is lower than the offspring's.

Algorithm 1 shows a basic pseudocode of the steady-state GA.

Algorithm 1. Steady-state GA algorithm basic structure

```
Input:A population
Output:An optimized population
Initialize population;
    while Termination criterion not satisfied do
    Select two parents from the population;
    Create two offspring using crossover and mutation;
    Evaluate the offspring with the Fitness function;
    Select two individuals in the population, which may be replaced by the offspring;
    Decide if this/these individuals will be replaced;
end
```

Concerning the fitness function, it must pursue both reduction and accuracy objectives when evaluating an IS chromosome, J. To do so, we will follow the proposal given in [7], where Cano et al. defined *AccRate* as the accuracy achieved by a K-NN classifier when classifying the entire training set using the currently selected subset as a reference and using leave-one-out as validation scheme

$$AccRate(J) = \mathrm{K-NNAccuracy}(J) \tag{12}$$

RedRate as the reduction rate achieved over the currently selected (maintained) instances

$$RedRate(J) = \frac{\#\mathrm{Instances\ Selected}(J)}{N} \tag{13}$$

and a real-valued weighting factor, α, to adjust the strength of each term in the resulting fitness value. Eq. (14) defines the full fitness function

$$Fitness(J) = \alpha \cdot AccRate(J) + (1-\alpha) \cdot RedRate(J) \tag{14}$$

Following the recommendations given in [7], EIS-RFS will employ a value $\alpha = 0.5$, which should offer an adequate trade-off between accuracy and reduction goals.

3.2. Selecting features by means of a fuzzy RST procedure

The concept of discernibility, defined in the realm of fuzzy RST, allows several useful approaches to data reduction to be developed [12,34]. The elements of a given data set can be analyzed, identifying which are discernible (with respect to the elements belonging to the other classes of the domain), regarding the specific set of features considered.

Therefore, a straightforward method to properly characterize a specific training set is to select those features which are able to fully discern all the instances of the training set (or, at least, discern them as much as possible). This way, the pruned training set can maintain its capabilities of separating instances belonging to different classes (or even increase them, by the removal of noisy and irrelevant features), while its size is reduced.

Eq. (11) gives a proper measure to evaluate the discernibility of a subset of features. The first step to compute this measure consists of defining a similarity measure between two different values of a same feature. This measure can be modeled as a fuzzy tolerance relation, R. For quantitative values, a suitable measure was defined in [36]:

$$R_a(x,y) = max\left(min\left(\left(\frac{a(y)-a(x)+\sigma_a}{\sigma_a}, \frac{a(x)-a(y)+\sigma_a}{\sigma_a}\right), 0\right)\right) \tag{15}$$

where x and y are two different instances belonging to the training set, and σ_a denotes the standard deviation of a. For nominal attributes, instead of using the equality metric:

$$R_a(x,y) = \begin{cases} 1 & \text{if } a(x) = a(y) \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

we propose the Value Difference Metric (VDM) [63,70], where two values are considered to be closer if they have more similar classifications (that is, more similar correlations with the output classes).

$$R_a(x,y) = \sum_{c=1}^{C} \left| \frac{N_{a,x,c}}{N_{a,x}} - \frac{N_{a,y,c}}{N_{a,y}} \right|^q \tag{17}$$

where:

- $N_{a,x}$ is the number of instances in the training set that have the same value as instance x for attribute a.
- $N_{a,x,c}$ is the number of instances in the training set that have the same value as instance x for attribute a and output class c.
- C is the number of output classes in the problem domain.
- q is a constant (typically 2).

Although the joint use of the overlap metric with crisp connectives would lead to an approach similar to those based on classical RST, the use of VDM is preferred, since it provides a soft similarity measure, suitable to be combined with the use of fuzzy connectives.

Eqs. (16) and (17) allows us to employ the fuzzy B-indiscernibility relation defined by Eq. (6), in which $\mathcal{T}$ represents the minimum t-norm, $\mathcal{T}(x,y) = min(x,y), x,y \in [0,1]$. Then, it is possible to compute the lower approximation of a fuzzy set A in $\mathcal{X}$ by means of a fuzzy tolerance relation R, by using Eq. (7) (employing, as implicator, $\mathcal{I}$, the Lukasiewicz one, $\mathcal{I}(x,y) = min(1, 1-x+y), x,y \in [0,1]$). With these definitions, it is possible to obtain the degree of inclusion of the instances of the training set in the fuzzy B-positive region, POS_B, and the gamma measure for B, γ_B.

Once a suitable measure of quality for a given subset of features, B, has been defined (γ_B), a search procedure to find the best possible subset can be carried out. Following the recommendations given in [12], the QUICKREDUCT heuristic [35] will be employed. Algorithm 2 shows its basic pseudocode.

Algorithm 2. QUICKREDUCT heuristic

```
Input:A set of instances
Output:A subset of features (B)
B ← {};
repeat
    T ← B,best ←-1;
    foreach a∈ $(\mathcal{A} \setminus B)$ do
        If $\gamma_{B\cup\{a\}}$> best then
            T ← $B \cup \{a\}$, best ←$\gamma_{B\cup\{a\}}$;
        end
    end
    B ← T;
until $\gamma_B \geqslant MaxGamma$;
```

Basically, QUICKREDUCT considers all the features in the domain of the problem and tries to add them to the candidate subset T. Features are added only if γ_B is improved. This hillclimbing procedure is continued until an established *MaxGamma* value (typically 1) is reached.

This filter method enables suitable subsets of features (with $\gamma_B = 1$) to be found quickly, which properly represent the information contained in the data set considered as the input parameter. Note that, when this method is used inside the framework of a larger algorithm, either subsets of instances or the entire training set can be considered.

3.3. Hybrid model for simultaneous IS and FS

Once the two basic tools considered for performing IS and FS have been defined, we can describe the hybrid model which composes our approach. Basically, it is a steady-state GA for IS where, every time a fixed number of evaluations has been spent, an RST based FS procedure is applied to modify the features considered during the search. Therefore, at any time only a single feature subset will be used in the whole search procedure. As the search progresses, this subset will be upgraded and adapted, to fit with the best subset of instances found.

Fig. 1 shows a flowchart representing its main steps: Initialization (Step 1), feature selection procedure (Step 4), Instance Selection procedure (Step 5), and Output (Step 7). The rest of the operations (Steps 2,3 and 6) control whether each of the former procedures should be carried out. The properties of each step are detailed as follows:

1. **Initialization**: The initialization procedure consists of the initialization of the chromosomes of the population, and the selection of the initial subset of features. The chromosomes, representing different subsets of instances, are initialized randomly (taking binary values, that is, in {0,1}). Regarding the initial subset of features, two different subsets are considered:

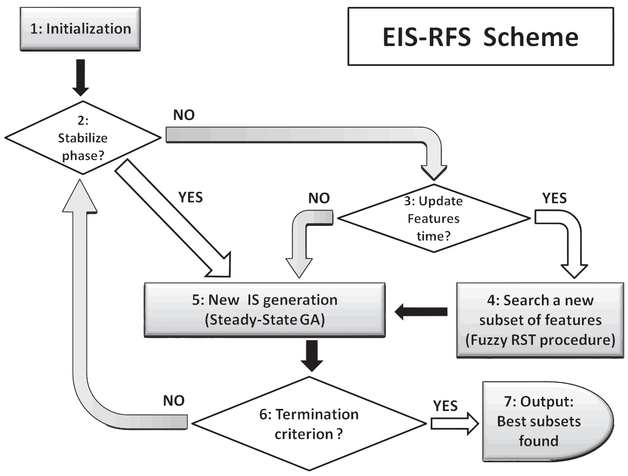

EIS-RFS Scheme
1: Initialization
2: Stabilize phase?
NO
YES
3: Update Features time?
NO
YES
4: Search a new subset of features (Fuzzy RST procedure)
5: New IS generation (Steady-State GA)
6: Termination criterion ?
NO
YES
7: Output: Best subsets found

Fig. 1. Flowchart depicting the main steps of EIS-RFS. Rectangles depict processes whereas rhombuses depict decisions taken by the algorithm.

- The full set of features of the domain.
- The subset of features selected by the RST based FS method using the whole training set as input.

The best performing subset (that is, the one which achieves the lowest error when applied to a K-NN classifier) is selected as the global subset of features of EIS-RFS. Only the features included in this subset will be considered in subsequent phases, until new changes in the subset are made (step 4).

2. **Stabilize phase**: Changes in the current subset of features are only considered if the search is not near its end. Therefore, if the current number of evaluations spent is higher than $\beta\cdot$ *MAX_EVALUATIONS* (usually with β near to 1), the stabilize stage is activated and no further changes in the subset of features selected are considered. This mechanism allows EIS-RFS to easily converge for hard problems, where the final subset of features is fixed before the end of the search. It allows EIS-RFS to focus its last efforts on optimizing the subsets of instances selected, performing a final refinement of the solutions achieved.
3. **Update Features**: If the Stabilize phase is not activated yet, it checks whether the FS procedure will be started or not. It will be performed every time *UpdateFS* evaluations have been spent by the steady-state GA.
4. **Search for a new subset of features**: This procedure consists of using the RST-based FS filter method, using as input the current best chromosome of the population (the best subset of instances found so far). The new subset of features obtained is tested by applying it to a K-NN classifier (considering as a reference set only the current best subset of instances). If this subset performs better than the former, it is accepted as the global subset of features of EIS-RFS.
5. **New IS generation**: An IS generation is carried out using the steady-state GA scheme described in Section 3.1. Note that, when evaluating a new chromosome, the K-NN classifier used in the fitness function will only consider the selected features in the global subset of features of EIS-RFS.
6. **Termination criterion**: The search process of EIS-RFS ends if the number of evaluations spent reaches the *MAX_EVALUATIONS* limit. Otherwise, a new cycle of the algorithm begins.
7. **Output: Best subsets found**: When the fixed number of evaluations runs out, the best chromosome of the population is selected as the best subset of instances found. The current global subset of selected features is designed as the best subset of features found. Both subsets are returned as the output of EIS-RFS.

The output of EIS-RFS (a subset of instances and a subset of features) defines a pruned version of the original training set. This set is ready to be used as a reference set by a K-NN classifier to perform a faster and more accurate classification of new test instances.

4. Experimental framework

This section presents the experimental study designed to test our proposal. Section 4.1 presents the classification data sets used throughout the study. Section 4.2 summarizes the algorithms selected for the comparison and the statistical procedures applied.

4.1. Data sets

To check the performance of EIS-RFS, we have selected a set of 43 classification data sets (30 standard data sets and 13 large data sets). These are well-known problems in the area, taken from the UCI Machine Learning Repository [18] and the

Table 1
Summary description for standard data sets.

Data Set	#Ex.	#Feat.	#Num.	#Nom.	#Cl.	Data set	#Ex.	#Feat.	#Num.	#Nom.	#Cl.
Australian	690	14	8	6	2	Iris	150	4	4	0	3
Balance	625	4	4	0	3	Led7digit	500	7	7	0	10
Bupa	345	6	6	0	2	Lymphography	148	18	3	15	4
Cleveland	303	13	13	0	5	Mammographic	961	5	5	0	2
Contraceptive	1,473	9	9	0	3	Monk-2	432	6	6	0	2
Crx	690	15	6	9	2	Newthyroid	215	5	5	0	3
Ecoli	336	7	7	0	8	Pima	768	8	8	0	2
Flare-solar	1,066	11	0	11	6	Saheart	462	9	8	1	2
German	1,000	20	7	13	2	Sonar	208	60	60	0	2
Glass	214	9	9	0	7	Spectheart	267	44	44	0	2
Haberman	306	3	3	0	2	Tic-tac-toe	958	9	0	9	2
Hayes-roth	160	4	4	0	3	Wine	178	13	13	0	3
Heart	270	13	13	0	2	Wisconsin	699	9	9	0	2
Hepatitis	155	19	19	0	2	Yeast	1,484	8	8	0	10
Housevotes	435	16	0	16	2	Zoo	101	16	0	16	7

Table 2
Summary description for large data sets.

Data Set	#Ex.	#Feat.	#Num.	#Nom.	#Cl.	Data Set	#Ex.	#Feat.	#Num.	#Nom.	#Cl.
Abalone	4,174	8	7	1	28	Satimage	6,435	36	36	0	7
Banana	5,300	2	2	0	2	Segment	2,310	19	19	0	7
Chess	3,196	36	0	36	2	Spambase	4,597	57	57	0	2
Marketing	8,993	13	13	0	9	Splice	3,190	60	0	60	3
Mushroom	8,124	22	0	22	2	Titanic	2,201	3	3	0	2
Page-blocks	5,472	10	10	0	5	Twonorm	7,400	20	20	0	2
Ring	7,400	20	20	0	2						

KEEL-dataset repository[1] [3,4]. Tables 1 and 2 summarizes their properties. For each data set, we provide its number of examples (#Ex.), Features (#Feat.) the number of numerical (#Num.) and nominal (#Nom.) attributes, and the number of classes (#Cl.). For continuous attributes, their values are normalized in the interval [0,1] to equalize the influence of attributes with different range domains.

The data sets considered are partitioned by using the ten fold cross-validation (10-fcv) procedure [64], that is, each data set is randomly partitioned into ten subsets, preserving the same size (the same number of instances) and the same class distribution between partitions. In an iterative process, one partition is selected as the test set whereas the training set is composed of the rest. Final results are obtained averaging the results obtained over the ten partitions (stochastic algorithms have been run three times).

4.2. Algorithms, parameters and statistical analysis

Several evolutionary preprocessing methods for the K-NN classifier have been selected to perform a comprehensive study of the capabilities of our approach. These methods are the following:

- **IS-SSGA**: A steady-state GA for IS. This evolutionary method has the same characteristics as EIS-RFS, but does not include any kind of feature selection process. Its behavior as an IS method was studied in depth in [7].
- **FS-SSGA**: A steady-state GA for FS. The features of the domain are encoded in the binary chromosomes, in a similar way to IS-SSGA. Note that in the fitness function of this method, the reduction rate is computed over the ratio of features selected. Therefore, its α weight must be very near to 1.0, to avoid an excessive deletion of features which may degrade the accuracy of the classifier too much.
- **IFS-SSGA**: A steady-state GA for simultaneous IS and FS. Its binary coded chromosomes encode both instances and features of the domain. The fitness function of this method only considers instances' reduction rate to compute the reduction ratio achieved.
- **FS-RST**: The fuzzy RST based feature selection method (FS-RST) used within the EIS-RFS framework, applied in isolation.
- **FS-RST + IS-SSGA**: FS-RST used as a preprocessor to IS-SSGA, that is, the preprocessed reference set obtained after the application of FS-RST is used as the input of the IS-SSGA method.
- **IS-SSGA + FS-RST**: IS-SSGA used as a preprocessor to FS-RST, that is, the preprocessed reference set obtained after the application of IS-SSGA is used as the input of the IS-SSGA method.

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

Table 3
Parameter specification for the algorithms tested in the experimentation.

Algorithm	Parameters (all K-NN based methods will use $k = 1$, unless explicitly stated in a particular experiment)
EIS-RFS	MAX_EVALUATIONS: 10000, Population size: 50, Crossover probability: 1.0, Mutation probability: 0.005 per bit, α: 0.5 *MaxGamma*: 1.0, UpdateFS: 100, β: 0.75
IS-SSGA	MAX_EVALUATIONS: 10000, Population size: 50, Crossover probability: 1.0, Mutation probability: 0.005 per bit, α: 0.5
FS-SSGA	MAX_EVALUATIONS: 10000, Population size: 50, Crossover probability: 1.0, Mutation probability: 0.005 per bit, α: 0.99
IFS-SSGA	MAX_EVALUATIONS: 10000, Population size: 50, Crossover probability: 1.0, Mutation probability: 0.005 per bit, α: 0.5
FS-RST	*MaxGamma*: 1.0

In order to estimate the classification accuracy, the preprocessed training sets obtained as a result of the application of these methods are tested by using a K-NN classifier.

Many different configurations can be established for each combination of domain and method. However, for the sake of a fair comparison, we have selected a fixed set of parameters for each method. Table 3 summarizes them.

Hypothesis testing techniques are used to contrast the experimental results and provide statistical support for the analysis [59]. Specifically, we use non-parametric tests, since the initial conditions that guarantee the reliability of the parametric tests may not be satisfied, causing the statistical analysis to lose credibility [16,23].

Throughout the study, two nonparametric tests for pairwise statistical comparisons of classifiers will be employed. The first one is the well-known Wilcoxon Signed-Ranks Test [69]. The second one is the Contrast Estimation of medians [22], which is very useful for estimating the difference between two algorithms' performance. We describe its detailed definition in Appendix A.

Further information about these tests and other statistical procedures specifically designed for use in the field of Machine Learning can be found at the SCI2S thematic public website on *Statistical Inference in Computational Intelligence and Data Mining*.[2]

5. Results and analysis

This section is devoted to presenting and analyzing the results achieved in several studies performed to test the behavior of EIS-RFS and compare it with several related techniques. To this end, Section 5.1 studies the behavior of EIS-RFS and K-NN as the number of neighbors selected grow. Section 5.2 shows a comparison between EIS-RFS and the rest of the comparison methods selected in standard data sets. Finally, Section 5.3 compares EIS-RFS when applied to large data sets.

5.1. EIS-RFS vs K-NN

One of the most critical issues of the K-NN classifier lies in the selection of its k parameter, that is, the number of neighbors considered in the decision rule. Usually, an odd number of them is preferred, since this way ties in the decision of the class membership are less likely to occur.

Concerning EIS-RFS, this is also a key issue, since the specific number of neighbors selected will affect the way in which the accuracy assigned to the chromosomes is estimated, thus modifying its behavior during the search.

To analyze this issue, we have classified the 30 standard data sets with EIS-RFS and K-NN, using 1, 3, 5 and 7 neighbors. Table 4 shows the average accuracy results achieved (the best results in each data set and category are highlighted in bold) and the number of times that each method achieves the best result. Table 5 presents the results of the Wilcoxon Signed-Ranks Test performed to contrast the results in each category.

The results show the validity of EIS-RFS as a preprocessing method for the K-NN classifier. Its average accuracy is improved in every category studied (from 1 to 7 neighbors), and the number of data sets in which EIS-RFS offers a best result is always greater. Moreover, the Wilcoxon Signed-Ranks Test confirms that differences between the methods are significant at the 0.05 significance level.

Fig. 2 depicts this comparison graphically. The dots symbolize the accuracy achieved in test phase by EIS-RFS and K-NN in a concrete data set (30 points are represented in each graph). A straight line splits the graph, exactly at the points where the accuracy measure of both classifiers is equal. Therefore, those points below (right) of the line represent data sets where EIS-RFS behaves better than K-NN, whereas those points above (left) of the line represent the opposite.

Clearly, EIS-RFS outperforms K-NN in every case, albeit the improvement achieved with the application of preprocessing diminishes as the number of neighbors considered is increased. Similarly to IS algorithms, setting a value of k greater than 1 for EIS-RFS decreases its sensitivity to noise, smoothing the decision boundaries [71]. Therefore, the enhancement obtained if the number of neighbors selected is high will be lower, although its application will still be beneficial if its results are compared with those obtained without preprocessing data (K-NN).

Finally, a second conclusion arrived at this study is that EIS-RFS behaves slightly better if only 1 neighbor is considered. Thus, we will fix the number of neighbors considered by it to 1 in the rest of the experimental study.

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

Table 4
Average accuracy rates obtained by EIS-RFS and K-NN considering different numbers of neighbors.

Data set	1 neighbor		3 neighbors		5 neighbors		7 neighbors	
	EIS-RFS	K-NN	EIS-RFS	K-NN	EIS-RFS	K-NN	EIS-RFS	K-NN
Australian	**85.66**	81.45	**85.91**	84.78	**84.98**	84.78	**85.65**	84.78
Balance	**85.92**	79.04	**85.74**	83.37	**86.88**	86.24	87.83	**88.48**
Bupa	**65.72**	61.08	**64.91**	60.66	**64.37**	61.31	**63.75**	62.53
Cleveland	**55.16**	53.14	**56.42**	54.44	**56.47**	55.45	**56.81**	56.45
Contraceptive	**45.42**	42.77	**46.85**	44.95	**47.59**	46.85	**49.16**	48.27
Crx	**84.93**	79.57	**84.49**	84.20	85.07	**85.51**	**85.80**	85.65
Ecoli	**82.14**	80.70	79.84	**80.67**	80.55	**81.27**	80.87	**82.45**
Flare-solar	**66.32**	55.54	**65.48**	55.07	**65.95**	57.04	**66.04**	63.89
German	**70.80**	70.50	**70.90**	69.60	**74.10**	71.80	**74.30**	72.20
Glass	67.35	**73.61**	65.36	**70.11**	64.90	**66.85**	65.10	**66.83**
Haberman	**71.56**	66.97	**72.60**	70.58	**72.49**	66.95	**72.19**	69.90
Hayes-roth	**80.86**	35.70	**74.98**	24.82	**67.98**	23.95	**62.59**	26.86
Heart	**80.74**	77.04	**79.56**	77.41	78.89	**80.74**	**80.37**	79.26
Hepatitis	**82.58**	82.04	83.13	**83.88**	83.04	**85.21**	83.29	**83.88**
Housevotes	**94.48**	91.24	**95.84**	94.01	**94.00**	93.31	**93.32**	93.09
Iris	**96.00**	93.33	**94.00**	**94.00**	**96.00**	**96.00**	95.33	**96.00**
Led7digit	**73.20**	40.20	**74.60**	45.20	**70.40**	41.40	**70.20**	43.40
Lymphography	**77.15**	73.87	**77.44**	77.39	**82.65**	79.44	**82.25**	81.49
Mammographic	**80.65**	76.38	**81.27**	79.19	**81.56**	81.06	**81.48**	81.17
Monk-2	**100.00**	77.91	**97.55**	96.29	**97.07**	94.75	**100.00**	89.16
Newthyroid	96.77	**97.23**	**95.41**	95.37	**94.91**	93.98	**96.32**	92.58
Pima	**74.80**	70.33	**73.45**	72.93	**73.72**	73.06	**74.50**	72.93
Saheart	**68.82**	64.49	**68.67**	68.18	**68.64**	67.10	**68.46**	66.45
Sonar	80.76	**85.55**	78.83	**83.07**	78.60	**83.10**	78.48	**80.21**
Spectfheart	**76.82**	69.70	**74.99**	71.20	**78.33**	71.97	**77.86**	77.58
Tic-tac-toe	**78.29**	73.07	**78.01**	77.56	78.12	**83.30**	78.69	**82.88**
Wine	**97.19**	95.52	94.35	**95.49**	**96.26**	96.05	96.35	**96.63**
Wisconsin	**96.42**	95.57	**96.33**	96.00	96.28	**96.57**	96.14	**97.00**
Yeast	**53.37**	50.47	**56.20**	53.17	55.86	**56.74**	**57.55**	57.49
Zoo	**96.39**	92.81	**94.81**	92.81	**94.97**	93.64	**94.64**	92.97
Average	**78.88**	72.89	**78.26**	74.55	**78.35**	75.18	**78.51**	75.75
Best result (of 30)	27	3	25	6	21	10	21	9

Table 5
Wilcoxon Signed-Ranks Test results for EIS-RFS vs K-NN.

EIS-RFS vs K-NN	R^+	R^-	P-value
1 neighbor	418.0	47.0	0.00004
3 neighbors	357.0	78.0	0.00182
5 neighbors	310.0	125.0	0.04552
7 neighbors	335.5	129.5	0.03363

5.2. Comparison with IS, FS and hybrid techniques

Table 6 shows the results measured by accuracy in test data for each method considered in the study. For each data set, the mean accuracy and the standard deviation are computed. The best case in each data set is highlighted in bold. The last row in each table shows the average considering all data sets.

On the other hand, Table 7 shows the average reduction rates achieved. In instances' search space, the reduction is defined by Eq. (13). The reduction rate for features is computed in a similar way, considering only those features selected (maintained) in the final reference set

$$RedRate_{\text{Features}} = \frac{\#\text{Features Selected}}{M} \tag{18}$$

In each category, only those methods that perform any kind of reduction are considered, that is:

- Instances' search space: EIS-RFS, IS-SSGA, IFS-SSGA, FS-RST + IS-SSGA, IS-SSGA + FS-RST.
- Features' search space: EIS-RFS, FS-SSGA, IFS-SSGA, FS-RST, FS-RST + IS-SSGA, IS-SSGA + FS-RST.

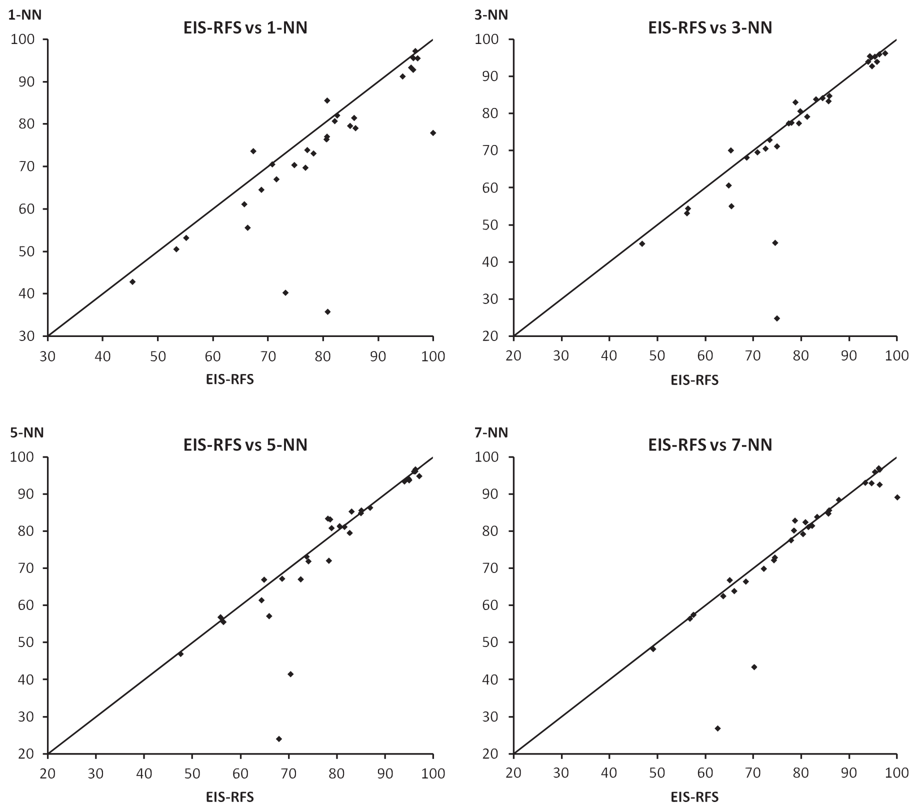


Fig. 2. Graphical comparison of EIS-RFS and K-NN using 1, 3, 5 and 7 neighbors. EIS-RFS shows a better performance in every case. However this improvement is lower as the number of neighbors increases.

Finally, Table 8 reports the time elapsed by each method in training phase (in seconds).[3] Note that running times in test phase are not reported due to the fact that they are too low to show interesting differences and efficiency in test phase is already reflected by the reduction rates achieved (the higher the reduction rates are, the less running time will be needed).

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

- EIS-RFS achieves the best accuracy result, obtaining the best overall result in 13 of the 30 data sets.
- Concerning reduction in instances' space, all the methods considered achieve a similar rate. Hence, the computational time spent by the K-NN classifier in test phase will be very low if any of the reference sets produced is used, including the one preprocessed by EIS-RFS.
- In the features' space, EIS-RFS shows a similar behavior to FS-RST. Although these reduction rates are low when compared with the ones achieved by the evolutionary techniques (FS-SSGA and IFS-SSGA), these methods could delete too much features due to the fitness function used. Note that, within the evolutionary search (and especially in its first steps), it is easier to remove features than to find an accurate combination of them in order to improve the fitness value. Thus, by sacrificing some of its reduction power, EIS-RFS is able to find better subsets of features, reinforcing our hypothesis of hybridizing Fuzzy-RST with EIS.
- Concerning the time elapsed, EIS-RFS spent a slightly higher time than the rest of methods, except for FS-RST (whose computational time is not comparable since it only evaluates a single reference set, instead of the 10,000 evaluations performed by the evolutionary methods), and FS-SSGA, whose time requirements are twice those of the rest.

Regarding the statistical analysis performed, Table 9 presents the results of the Wilcoxon Signed Ranks test, while Table 10 shows the results of the Contrast Estimation. Note that these tests have been carried out considering the average accuracy of the results obtained.

The Wilcoxon test shows that our approach, EIS-RFS, statistically outperforms all the comparison methods with a level of significance $\alpha = 0.01$; that is, no comparison of EIS-RFS and any comparison method achieves a P-value equal to or higher than 0.01. This is a strong result, which supports the fact that EIS-RFS clearly outperforms all the other techniques.

[3] The experiments have been carried out on a machine with a Dual Core 3,20 GHz processor and 2 GB of RAM, running under the Fedora 4 operating System.

Table 6
Accuracy results in test phase.

Data set	EIS-RFS	IS-SSGA	FS-SSGA	IFS-SSGA	FS-RST	FS-RST+IS-SSGA	IS-SSGA+ FS-RST
Australian	**85.66** ± 2.27	85.65 ± 2.77	85.07 ± 3.49	85.36 ± 3.31	81.45 ± 4.52	85.51 ± 2.92	80.87 ± 4.49
Balance	85.92 ± 2.62	**86.40** ± 3.08	70.89 ± 9.80	84.31 ± 4.85	79.04 ± 6.81	71.89 ± 3.28	61.05 ± 6.82
Bupa	**65.72** ± 8.79	61.14 ± 9.37	59.91 ± 10.19	62.72 ± 8.40	62.51 ± 7.78	61.14 ± 9.60	61.14 ± 7.17
Cleveland	55.16 ± 5.82	52.82 ± 4.47	51.47 ± 9.47	**56.13** ± 6.07	52.51 ± 9.49	54.42 ± 5.07	51.51 ± 9.37
Contraceptive	**45.42** ± 5.14	44.54 ± 4.61	41.96 ± 3.57	45.15 ± 2.32	42.63 ± 3.73	44.54 ± 4.92	45.15 ± 4.07
Crx	**84.93** ± 5.72	84.64 ± 4.22	81.16 ± 7.61	84.64 ± 5.08	81.30 ± 6.28	83.19 ± 4.79	80.72 ± 5.82
Ecoli	**82.14** ± 8.42	80.38 ± 5.69	78.90 ± 7.30	77.70 ± 5.52	76.58 ± 14.73	77.18 ± 6.49	75.97 ± 15.00
Flare-solar	66.32 ± 2.94	64.82 ± 3.37	62.76 ± 3.65	**67.35** ± 4.12	63.23 ± 5.56	64.82 ± 3.53	63.04 ± 5.31
German	**70.80** ± 4.24	70.40 ± 3.24	69.50 ± 2.68	70.10 ± 3.48	67.90 ± 3.41	69.20 ± 3.34	69.60 ± 3.20
Glass	67.35 ± 11.83	67.10 ± 14.74	71.80 ± 14.30	71.23 ± 10.64	**74.50** ± 13.17	67.10 ± 15.68	66.12 ± 13.66
Haberman	71.56 ± 7.34	71.23 ± 5.40	72.81 ± 6.15	**72.83** ± 5.99	65.68 ± 6.58	71.23 ± 6.20	71.23 ± 6.42
Hayes–roth	80.86 ± 11.70	69.15 ± 11.69	**83.93** ± 8.33	79.80 ± 11.65	76.07 ± 14.07	74.87 ± 11.13	77.95 ± 12.76
Heart	80.74 ± 6.34	81.11 ± 7.90	76.67 ± 6.06	**82.59** ± 6.31	78.89 ± 6.77	79.26 ± 8.84	71.48 ± 6.93
Hepatitis	**82.58** ± 7.99	79.33 ± 8.71	76.21 ± 7.89	80.67 ± 6.13	79.50 ± 7.95	80.04 ± 8.38	71.13 ± 7.52
Housevotes	**94.48** ± 3.67	93.79 ± 3.43	94.01 ± 4.53	94.46 ± 4.37	90.78 ± 6.47	74.42 ± 4.04	62.56 ± 5.99
Iris	**96.00** ± 4.92	94.67 ± 2.81	95.33 ± 4.50	94.67 ± 4.22	93.33 ± 5.44	94.67 ± 3.32	94.67 ± 5.73
Led7digit	73.20 ± 4.99	**73.40** ± 2.84	63.00 ± 6.94	71.40 ± 4.81	63.60 ± 5.87	17.40 ± 2.99	17.00 ± 6.15
Lymphography	77.15 ± 12.15	77.92 ± 9.39	**78.49** ± 9.12	74.92 ± 10.79	77.38 ± 11.21	78.06 ± 9.87	66.55 ± 12.18
Mammographic	**80.65** ± 4.51	79.50 ± 3.85	75.86 ± 6.07	80.15 ± 6.23	75.76 ± 4.97	79.50 ± 3.85	79.50 ± 5.08
Monk-2	**100.00** ± 0.00	83.53 ± 6.21	**100.00** ± 0.00	98.64 ± 3.07	77.91 ± 5.71	**100.00** ± 5.95	96.13 ± 5.50
Newthyroid	96.77 ± 4.83	98.16 ± 3.20	96.30 ± 1.95	96.32 ± 3.60	97.23 ± 2.39	98.16 ± 3.58	**98.61** ± 2.23
Pima	**74.80** ± 3.71	72.26 ± 4.44	67.70 ± 4.59	73.83 ± 3.15	70.33 ± 3.71	72.26 ± 4.48	72.26 ± 3.67
Saheart	68.82 ± 7.16	**69.27** ± 3.70	61.24 ± 3.91	67.99 ± 5.69	64.49 ± 4.21	65.39 ± 4.09	66.04 ± 4.61
Sonar	80.76 ± 7.88	75.45 ± 11.74	**84.62** ± 8.65	75.50 ± 12.59	81.69 ± 9.83	71.10 ± 11.61	69.21 ± 9.62
Spectfheart	76.82 ± 7.07	75.31 ± 5.96	74.17 ± 6.34	75.34 ± 7.31	70.04 ± 8.00	**79.12** ± 5.92	70.10 ± 8.46
Tic-tac-toe	78.29 ± 5.07	78.71 ± 3.36	**83.51** ± 3.10	77.87 ± 5.25	73.07 ± 2.70	65.35 ± 3.44	65.35 ± 2.63
Wine	**97.19** ± 5.09	92.68 ± 7.91	94.90 ± 3.30	94.93 ± 3.17	95.49 ± 4.40	96.05 ± 9.24	83.17 ± 4.38
Wisconsin	96.42 ± 1.55	96.13 ± 2.95	95.14 ± 2.62	95.86 ± 2.47	95.57 ± 2.73	**96.71** ± 3.36	95.99 ± 2.72
Yeast	53.37 ± 3.36	**54.18** ± 4.38	52.30 ± 3.94	53.50 ± 3.77	52.23 ± 4.39	**54.18** ± 4.74	53.30 ± 4.02
Zoo	96.39 ± 4.80	94.22 ± 7.94	95.42 ± 6.00	90.72 ± 7.09	**96.50** ± 4.61	89.14 ± 7.88	86.08 ± 4.18
Average	**78.88** ± 5.73	76.93 ± 5.78	76.50 ± 5.87	77.89 ± 5.72	75.24 ± 6.58	73.86 ± 6.17	70.78 ± 6.52
Best result (of 30)	13	4	5	4	2	4	1

Furthermore, median estimators computed by the Contrast Estimation (Table 10) represent the quantitative difference between the different methods considered in the experimental study. Fig. 3 depicts these results graphically. As can be seen in the graph, EIS-RFS achieves a moderate improvement on IFS-SSGA. This improvement is greater when comparing EIS-RFS with IS-SSGA, FS-RST + IS-SSGA and FS-SSGA. The greatest differences are found comparing it with FS-RST and IS-SSGA + FS-RST.

In summary, all these results depict EIS-RFS as an outstanding approach for enhancing the behavior of the K-NN classifier, with respect to the related techniques selected. It offers the most accurate results, while reduction rates are maintained with respect to its related techniques (thus the test phase will take the same computational resources – time and storage requirement – as the rest of methods). Moreover, its computational time in training phase is comparable to the rest of the evolutionary techniques, allowing it to be employed in standard classification problems with ease.

In addiction, it is important to point out that it significantly improves the non-hybrid proposals considered in the experimental framework: FS-RST + IS-SSGA and IS-SSGA + FS-RST. This fact reinforces the validity of the hybrid approach, in contrast to simply using both techniques one after the other.

5.3. Comparison in large domains

Table 11 shows the results measured by accuracy in test data for each method considered in the study with large data sets (including the 1-NN classifier as baseline). For each one, the mean accuracy and the standard deviation are computed. The best case in each data set is highlighted in bold. The last row in each table shows the average considering all data sets.

Tables 12 and 13 reports the average reduction rates achieved and the time elapsed footnote 3, respectively.

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

- EIS-RFS again achieves the best accuracy result, obtaining the best overall result in 11 of the 13 data sets. In addition, it still greatly outperforms the 1-NN classifier.
- Concerning reduction in instances' space, all the methods considered achieve a similar rate. Again, the computational time spent by the K-NN classifier in test phase will be very low if any of the reference sets produced is used, including the one preprocessed by EIS-RFS.

Table 7
Average reduction results over instances and features.

Data set	Instances					Features					
	EIS-RFS	IS-SSGA	IFS-SSGA	FS-RST + IS-SSGA	IS-SSGA + FS-RST	EIS-RFS	FS-SSGA	IFS-SSGA	FS-RST	FS-RST + IS-SSGA	IS-SSGA + FS-RST
Australian	0.8872	0.8799	0.8808	0.8808	0.8799	0.1571	0.8071	0.7929	0.0000	0.7929	0.2643
Balance	0.8464	0.8686	0.8085	0.9085	0.8686	0.0000	0.3000	0.0000	0.0000	0.0000	0.5250
Bupa	0.8502	0.8644	0.8644	0.8644	0.8644	0.0000	0.3667	0.4333	0.1274	0.4333	0.0000
Cleveland	0.9014	0.9171	0.9289	0.9289	0.9171	0.0462	0.7385	0.6077	0.3908	0.6077	0.5231
Contraceptive	0.7637	0.7530	0.7530	0.7530	0.7530	0.0667	0.4556	0.5889	0.0360	0.5889	0.1222
Crx	0.8914	0.8816	0.8805	0.8805	0.8816	0.1800	0.5667	0.5533	0.2000	0.5533	0.4067
Ecoli	0.8882	0.9077	0.9130	0.9130	0.9077	0.1286	0.1714	0.1857	0.2286	0.1857	0.2571
Flare-solar	0.8122	0.8391	0.8005	0.8405	0.8391	0.0556	0.5111	0.5778	0.1556	0.5778	0.3000
German	0.8014	0.7914	0.7928	0.7928	0.7914	0.2350	0.5150	0.7450	0.1450	0.7450	0.4900
Glass	0.8718	0.8791	0.8791	0.8791	0.8791	0.0444	0.4444	0.4556	0.0168	0.4556	0.0778
Haberman	0.9306	0.9379	0.9379	0.9379	0.9379	0.0000	0.6667	0.5333	0.0254	0.5333	0.0000
Hayes-roth	0.8544	0.8384	0.8452	0.8452	0.8384	0.2500	0.2500	0.2500	0.1000	0.2500	0.2500
Heart	0.9255	0.9506	0.9230	0.9230	0.9506	0.2308	0.4538	0.5692	0.1846	0.5692	0.6000
Hepatitis	0.9262	0.9226	0.9355	0.9355	0.9226	0.5368	0.6684	0.5421	0.4263	0.5421	0.7211
Housevotes	0.9387	0.9410	0.9653	0.9653	0.9410	0.3500	0.7000	0.7313	0.0188	0.7313	0.8625
Iris	0.9511	0.9481	0.9481	0.9481	0.9481	0.1250	0.4000	0.4500	0.0000	0.4500	0.0000
Led7digit	0.9416	0.9071	0.9491	0.9491	0.9071	0.0000	0.0143	0.0000	0.0143	0.0000	0.8571
Lymphography	0.9257	0.8994	0.9234	0.9234	0.8994	0.4444	0.6500	0.6500	0.2611	0.6500	0.6944
Mammographic	0.8322	0.8229	0.7829	0.8229	0.8229	0.0000	0.5000	0.6200	0.3396	0.6200	0.0000
Monk-2	0.9342	0.8570	0.9406	0.9406	0.8570	0.5000	0.5000	0.5333	0.0000	0.5333	0.5000
Newthyroid	0.9473	0.9571	0.9571	0.9571	0.9571	0.0600	0.3000	0.3800	0.0000	0.3800	0.1000
Pima	0.7911	0.8187	0.8187	0.8187	0.8187	0.0000	0.5750	0.4375	0.0000	0.4375	0.0875
Saheart	0.8668	0.8841	0.8778	0.8778	0.8841	0.0000	0.6333	0.5778	0.0000	0.5778	0.3111
Sonar	0.8899	0.8595	0.8974	0.8974	0.8595	0.2900	0.6633	0.6600	0.7183	0.6600	0.9167
Spectfheart	0.9497	0.9426	0.9409	0.9409	0.9426	0.2727	0.6750	0.6614	0.2750	0.6614	0.8773
Tic-tac-toe	0.8655	0.7917	0.8047	0.8747	0.7917	0.0000	0.2444	0.2889	0.0000	0.2889	0.8889
Wine	0.9451	0.9538	0.9557	0.9557	0.9538	0.3308	0.4538	0.4538	0.5231	0.4538	0.7462
Wisconsin	0.9103	0.9027	0.9048	0.9048	0.9027	0.0444	0.3889	0.3222	0.0000	0.3222	0.3667
Yeast	0.7550	0.7485	0.7485	0.7485	0.7485	0.0375	0.0875	0.1625	0.1256	0.1625	0.2375
Zoo	0.8634	0.8714	0.8468	0.8468	0.8714	0.2125	0.7125	0.3750	0.2750	0.3750	0.7500
Average	0.8819	0.8779	0.8802	0.8885	0.8779	0.1533	0.4804	0.4713	0.1529	0.4713	0.4244

- In the features' space, EIS-RFS obtains a slightly lower result than the rest of the reference techniques. However, with the high reduction rates obtained in the instances' space, the lower reduction rates over features will not cause the K-NN classifier to consume too much time in test phase.
- Concerning time elapsed, we again obtain the same results: EIS-RFS spent a slightly higher time than the rest of the methods, except for FS-RST and 1-NN (whose computational time is not comparable since they only evaluate a single reference set, instead of the 10,000 evaluations performed by the evolutionary methods), and FS-SSGA, whose time requirements are greater than the rest.

This time, the statistical analysis performed is shown in Tables 14 and 15. The former presents the results of the Wilcoxon Signed Ranks test, the latter shows the results of the Contrast Estimation. Again, note that these tests have been carried out considering the average accuracy results obtained.

The Wilcoxon test shows that our approach, EIS-RFS, statistically outperforms all the comparison methods with a level of significance $\alpha = 0.01$; that is, no comparison of EIS-RFS and any comparison method achieves a P-value equal to or higher than 0.01. This result confirms that the good behavior of EIS-RFS is maintained when applied to large data sets.

Median estimators computed by Contrast Estimation (Table 15) represent greater differences than in the former study, although similar conclusions can be drawn. Fig. 4 depicts these results graphically. As can be seen in the graph, EIS-RFS achieves a moderate improvement over the purely evolutionary methods (IS-SSGA, FS-SSGA and IFS-SSGA), which is greater when compared with FS-RST. The greatest differences are found comparing it with and FS-RST + IS-SSGA, IS-SSGA + FS-RST and 1-NN.

The results obtained in this part of the study contrast the quality of EIS-RFS further. Its capabilities are not diminished if it is applied over large data sets, even maintaining the same number of evaluations as in the standard study. Moreover, although its application in these domains is costly, the computational times reported suggest that EIS-RFS can be used in these domains without the necessity of exceptional computer resources, allowing the model to be considered for use in real-world applications.

Table 8
Average time elapsed (training phase), in seconds.

Data set	EIS-RFS	IS-SSGA	FS-SSGA	IFS-SSGA	FS-RST	FS-RST + IS-SSGA	IS-SSGA + FS-RST
Australian	82.54	79.16	161.39	48.14	0.70	81.13	79.90
Balance	54.44	38.71	88.56	38.33	0.03	29.41	37.45
Bupa	20.71	13.70	32.65	11.33	0.04	13.17	13.28
Cleveland	19.29	11.89	33.37	9.04	0.10	10.17	11.93
Contraceptive	316.30	348.66	704.06	306.52	1.32	347.29	339.26
Crx	86.38	79.72	220.59	70.56	0.46	83.90	83.55
Ecoli	20.68	10.92	37.50	11.17	0.05	10.35	11.06
Flare-solar	183.44	160.00	349.09	123.76	0.01	146.04	145.15
German	304.94	252.59	591.00	167.51	2.07	245.40	263.49
Glass	10.30	5.39	15.93	5.18	0.05	5.63	5.56
Haberman	9.41	7.09	13.63	6.06	0.01	7.15	7.07
Hayes-roth	3.86	2.68	5.00	2.52	0.02	2.74	2.77
Heart	14.57	8.03	32.98	7.01	0.06	7.91	8.02
Hepatitis	8.50	3.83	13.08	3.21	0.04	3.43	3.90
Housevotes	39.42	24.98	82.91	17.38	0.02	15.94	25.31
Iris	4.40	2.44	5.22	2.33	0.02	2.55	2.45
Led7digit	40.50	25.05	88.31	28.87	0.00	17.60	25.21
Lymphography	8.14	3.97	11.77	3.23	0.02	3.19	3.99
Mammographic	127.75	116.67	205.34	77.57	0.20	105.70	112.27
Monk-2	27.92	20.72	46.18	14.22	0.02	14.06	20.24
Newthyroid	8.03	3.68	11.76	3.63	0.01	3.81	3.66
Pima	96.68	85.38	175.95	72.09	0.29	77.98	78.76
Saheart	33.45	25.98	62.64	22.45	0.15	24.01	25.78
Sonar	136.71	17.12	66.79	13.86	0.40	9.91	17.22
Spectfheart	40.33	15.14	80.53	15.25	0.27	12.10	15.23
Tic-tac-toe	176.75	150.31	348.57	132.37	0.06	75.92	138.99
Wine	7.67	3.62	14.80	3.34	0.03	3.40	3.64
Wisconsin	73.61	55.66	159.05	50.62	0.10	49.54	53.67
Yeast	420.58	354.55	825.97	364.96	1.47	351.06	342.82
Zoo	5.31	2.52	5.12	2.46	0.01	2.33	2.46
Average	79.42	64.34	149.66	54.50	0.27	58.76	62.80

Table 9
Wilcoxon signed-ranks test results.

Comparison	R^+	R^-	*P*-value
EIS-RFS vs IS-SSGA	381	84	0.00158
EIS-RFS vs FS-SSGA	342	93	0.00599
EIS-RFS vs IFS-SSGA	371.5	93.5	0.00335
EIS-RFS vs FS-RST	426	39	0.00001
EIS-RFS vs FS-RST + IS-SSGA	389	39	0.00007
EIS-RFS vsIS-SSGA + FS-RST	456	39	0.00000

Table 10
Contrast estimation results.

EIS-RFS vs	Median estimation
IS-SSGA	1.031
FS-SSGA	1.946
IFS-SSGA	0.435
FS-RST	2.941
FS-RST + IS-SSGA	1.706
IS-SSGA + FS-RST	3.966

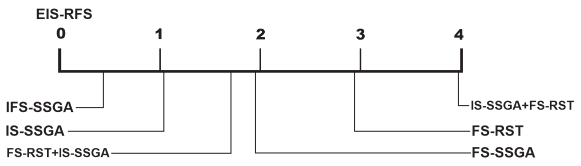


Fig. 3. Graphic depicting the Contrast Estimation of medians between EIS-RFS and the comparison methods.

Table 11
Accuracy results in test phase.

Data set	EIS-RFS	IS-SSGA	FS-SSGA	IFS-SSGA	FS-RST	FS-RST + IS-SSGA	IS-SSGA + FS-RST	1-NN
Abalone	**26.31** ± 2.01	21.78 ± 1.96	20.82 ± 1.98	20.44 ± 1.81	19.84 ± 1.89	19.66 ± 2.01	12.87 ± 2.17	10.38 ± 1.74
Banana	**89.34** ± 2.70	88.42 ± 0.95	87.49 ± 1.05	88.06 ± 1.38	87.49 ± 1.05	81.70 ± 2.78	78.48 ± 2.88	74.76 ± 2.08
Chess	**91.30** ± 5.38	86.83 ± 2.36	64.23 ± 0.76	67.03 ± 1.06	75.47 ± 2.07	73.23 ± 5.79	74.86 ± 5.65	68.99 ± 4.84
Marketing	**30.12** ± 1.85	27.10 ± 1.15	27.02 ± 1.14	29.09 ± 1.70	27.58 ± 1.06	24.90 ± 2.00	18.35 ± 1.87	17.20 ± 1.52
Mushroom	**100.00** ± 0.00	99.96 ± 0.08	**100.00** ± 0.00	99.94 ± 0.13	98.52 ± 0.35	98.26 ± 0.55	99.00 ± 0.34	**100.00** ± 0.00
Page-blocks	95.25 ± 5.51	94.97 ± 0.83	**96.44** ± 0.73	95.63 ± 0.44	95.07 ± 0.76	95.10 ± 5.72	85.48 ± 6.01	76.73 ± 5.84
Ring	**86.01** ± 1.86	74.93 ± 1.47	83.01 ± 1.38	82.01 ± 1.08	79.18 ± 1.35	71.39 ± 1.98	61.39 ± 1.94	50.25 ± 1.64
Satimage	**89.60** ± 1.87	87.73 ± 1.65	89.44 ± 1.96	88.65 ± 1.68	85.21 ± 1.99	84.36 ± 1.95	83.30 ± 1.97	88.38 ± 1.63
Segment	94.37 ± 0.94	94.59 ± 1.46	**96.84** ± 1.02	95.93 ± 1.21	94.29 ± 1.61	94.92 ± 1.03	93.49 ± 0.99	96.06 ± 0.82
Spambase	**89.35** ± 3.05	82.60 ± 0.72	83.47 ± 1.40	87.54 ± 2.07	81.74 ± 1.74	76.93 ± 3.23	79.43 ± 3.20	77.89 ± 2.49
Splice	**83.07** ± 2.32	73.57 ± 1.36	76.93 ± 2.61	72.95 ± 2.49	85.89 ± 1.85	72.46 ± 2.38	58.26 ± 2.40	60.55 ± 1.85
Titanic	**79.38** ± 11.23	78.78 ± 2.33	76.10 ± 5.60	78.74 ± 2.50	59.16 ± 8.08	51.35 ± 11.61	61.44 ± 11.94	13.98 ± 12.33
Twonorm	**96.54** ± 1.34	95.54 ± 1.64	94.86 ± 1.76	94.59 ± 1.83	77.86 ± 1.72	80.73 ± 1.37	81.10 ± 1.47	89.35 ± 1.46
Average	**80.82** ± 3.08	77.45 ± 1.38	76.67 ± 1.65	76.97 ± 1.49	74.41 ± 1.96	71.15 ± 3.26	68.27 ± 3.29	63.42 ± 2.94
Best result (of 13)	11	0	3	0	0	0	0	1

Table 12
Average reduction results over instances and features.

Data set	Instances					Features					
	EIS-RFS	IS-SSGA	IFS-SSGA	FS-RST + IS-SSGA	IS-SSGA + FS-RST	EIS-RFS	FS-SSGA	IFS-SSGA	FS-RST	FS-RST + IS-SSGA	IS-SSGAFS-RST
Abalone	0.7401	0.7426	0.7391	0.6913	0.7426	0.3575	0.5875	0.4625	0.5000	0.5000	0.5000
Banana	0.7509	0.7560	0.7483	0.7623	0.7560	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
Chess	0.7466	0.7578	0.7686	0.7184	0.7578	0.4722	0.4333	0.5306	0.9444	0.9444	0.5820
Marketing	0.7420	0.7357	0.7341	0.7326	0.7357	0.3585	0.7000	0.6385	0.3846	0.3846	0.3639
Mushroom	0.7620	0.7619	0.7609	0.7244	0.7619	0.8383	0.8091	0.5955	0.9545	0.9545	0.9121
Page-blocks	0.7643	0.7661	0.7610	0.7631	0.7661	0.7922	0.6000	0.4300	0.7000	0.7000	0.7000
Ring	0.7308	0.7258	0.7344	0.7345	0.7258	0.4537	0.5050	0.5450	0.7000	0.7000	0.6350
Satimage	0.7498	0.7528	0.7461	0.7020	0.7528	0.5700	0.5556	0.4611	0.8611	0.8611	0.6140
Segment	0.8062	0.8053	0.7993	0.7731	0.8053	0.0000	0.6895	0.6737	0.0000	0.0000	0.0000
Spambase	0.7549	0.7538	0.7389	0.7349	0.7538	0.5584	0.5298	0.5930	0.8596	0.8596	0.6930
Splice	0.7572	0.7431	0.7428	0.6913	0.7431	0.6833	0.8317	0.8317	0.8667	0.8667	0.8333
Titanic	0.8299	0.8283	0.8202	0.8156	0.8283	0.0000	0.5333	0.0333	0.0000	0.0000	0.0000
Twonorm	0.7521	0.7557	0.7285	0.7194	0.7557	0.0000	0.0000	0.0100	0.7500	0.7500	0.7500
Average	0.7605	0.7604	0.7556	0.7356	0.7604	0.3911	0.5211	0.4465	0.5785	0.5785	0.5064

Table 13
Average time elapsed (training phase), in seconds.

Data set	EIS-RFS	IS-SSGA	FS-SSGA	IFS-SSGA	FS-RST	FS-RST + IS-SSGA	IS-SSGA + FS-RST
Abalone	3739	3546	7992	3265	27	3391	3436
Banana	2496	3145	6992	3343	2	2948	3099
Chess	14345	5614	20910	4794	1	5249	5456
Marketing	21471	27793	47621	19468	98	25989	27774
Mushroom	33249	33121	62833	23482	4	31833	33136
Page-blocks	5853	7942	16681	6668	14	7419	7741
Ring	26797	24125	59894	20742	325	23185	24401
Satimage	62529	26160	82022	27992	426	25574	27607
Segment	1384	1641	4891	1206	4	1567	1618
Spambase	41759	19965	63763	19872	560	18870	19472
Splice	62673	9431	26504	5898	395	8951	9376
Titanic	413	605	1402	516	0	573	603
Twonorm	24515	22273	70736	25004	292	20941	21865
Average	23171	14259	36326	12481	165	13576	14276

Table 14
Wilcoxon signed-ranks test results (large data sets).

Comparison	R^+	R^-	*P*-value
EIS-RFS vs IS-SSGA	89	2	0.00073
EIS-RFS vs FS-SSGA	71	7	0.00928
EIS-RFS vs IFS-SSGA	82	9	0.00806
EIS-RFS vs FS-RST	85	6	0.00342
EIS-RFS vs FS-RST + IS-SSGA	89	2	0.00073
EIS-RFS vs IS-SSGA + FS-RST	91	0	0.00024
EIS-RFS vs 1-NN	76	2	0.00146

Table 15
Contrast estimation results.

EIS-RFS vs	Median estimation
IS-SSGA	2.653
FS-SSGA	2.849
IFS-SSGA	2.598
FS-RST	4.855
FS-RST + IS-SSGA	7.260
IS-SSGA + FS-RST	10.784
1-NN	13.540

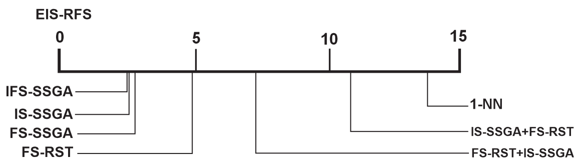


Fig. 4. Graphic depicting the Contrast Estimation of medians between EIS-RFS and the comparison methods in large data sets.

6. Conclusions

In this paper, we have presented EIS-RFS, a novel approach which introduces the cooperation between two well-known techniques for data reduction: A steady-state GA for IS, and a fuzzy RST based method for FS. The judicious selection of features performed by the RST based method has been shown to be beneficial for the search procedure performed by the GA, thus allowing our approach to achieve highly reduced training sets which greatly enhances the behavior of the K-NN classifier.

The experimental study performed has highlighted these improvements, especially in the case of 1-NN. It has also shown that EIS-RFS outperforms several preprocessing methods related to IS and FS, including hybrid models, considering classification accuracy. Moreover, these results are maintained when it is applied to higher size data sets, thus confirming the capabilities of our approach as a suitable preprocessing method for most of the standard problems present in supervised classification.

As regards future work, we point out the possibility of using EIS-RFS to enhance other kinds of machine learning algorithms. Given the nature of our method (in essence, a wrapper-based model) and the generality of the fuzzy rough feature selection method and the fitness function defined, it is possible to apply it to improve the results of the majority of machine learning methods. The only requirement would be the existence of a way in which the current performance of the method could be evaluated such as, for example, the accuracy measure of the K-NN classifier. A suitable starting point for this line would be the research shown in [8,9,37], where IS (namely TSS) is used to enhance other models such as decision trees, subgroup discovery methods and neural networks, respectively.

Acknowledgements

This work was supported by Project TIN2008-06681-C06-01. J. Derrac holds an FPU scholarship from Spanish Ministry of Education. Chris Cornelis would like to thank the Research Foundation – Flanders for funding his research.

Appendix A. A nonparametric method for analyzing medians of classifiers: the contrast estimation

The Contrast Estimation based on medians [22] can be used to estimate the difference between two classifiers' performance. It assumes that the expected differences between performances of algorithms are the same across data sets. Therefore, the performance of methods is reflected by the magnitudes of the differences between them in each domain.

The interest of this test lies in estimating the contrast between medians of samples of results considering all pairwise comparisons. The test obtains a quantitative difference computed through medians between two algorithms over multiple data sets, proceeding as follows:

1. For every pair of k algorithms in the experiment, compute the difference between the performances x of the two algorithms in each of the n data sets. That is, compute the differences

$$D_{i(u,v)} = x_{iu} - x_{iv} \tag{A.1}$$

where $i = 1,\ldots,n$; $u = 1,\ldots,k$; $v = 1,\ldots,k$. (consider only performance pairs where $u < v$).

2. Find the median of each set of differences (Z_{uv}, which can be regarded as the *unadjusted estimator* of the medians of the methods u and v, $M_u - M_v$). Since $Z_{uv} = Z_{vu}$, it is only required to compute Z_{uv} in those cases where $u < v$. Also note that $Z_{uu} = 0$.
3. Compute the mean of each set of unadjusted medians having the same first subscript, m_u:

$$m_u = \frac{\sum_{j=1}^{k} Z_{uj}}{k}, u = 1,\ldots,k \tag{A.2}$$

4. The estimator of $M_u - M_v$ is $m_u - m_v$, where u and v range from 1 through k. For example, the difference between M_1 and M_2 is estimated by $m_1 - m_2$

These estimators can be understood as an advanced global performance measure. Although this test cannot provide a probability of error associated with the rejection of the null hypothesis of equality, it is especially useful to estimate how far a method outperforms another one.

An implementation of the Contrast Estimation procedure can be found in the CONTROLTEST package, which can be obtained at the SCI2S thematic public website on *Statistical Inference in Computational Intelligence and Data Mining* (http://sci2s.ugr.es/sicidm/).

References

[1] D.W. Aha (Ed.), Lazy Learning, Springer, 2009.
[2] H. Ahn, K. Kim, Bankruptcy prediction modeling with hybrid case-based reasoning and genetic algorithms approach, Applied Soft Computing 9 (2009) 599–607.
[3] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera, Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework, Journal of Multiple-Valued Logic and Soft Computing 17 (2011).
[4] J. Alcalá-Fdez, L. Sánchez, S. García, M.J. del Jesus, S. Ventura, J.M. Garrell, J. Otero, C. Romero, J. Bacardit, V.M. Rivas, J.C. Fernández, F. Herrera, KEEL: a software tool to assess evolutionary algorithms for data mining problems, Soft Computing 13 (2008) 307–318.
[5] E. Alpaydin, Introduction to Machine Learning, second ed., The MIT Press, 2010.
[6] G. Bell, T. Hey, A. Szalay, Beyond the data deluge, Science 323 (2009) 1297–1298.
[7] 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.
[8] 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.
[9] J.R. Cano, F. Herrera, M. Lozano, S. García, Making cn2-sd subgroup discovery algorithm scalable to large size data sets using instance selection, Expert Systems with Applications 35 (2008) 1949–1965.

[10] D. Chen, C. Wang, Q. Hu, A new approach to attribute reduction of consistent and inconsistent covering decision systems with covering rough sets, Information Sciences 177 (2007) 3500v–3518.

[11] Y. Chen, E.K. Garcia, M.R. Gupta, A. Rahimi, L. Cazzanti, Similarity-based classification: Concepts and algorithms, Journal of Machine Learning Research 10 (2009) 747–776.

[12] C. Cornelis, R. Jensen, G. Hurtado, D. Slezak, Attribute selection with fuzzy decision reducts, Information Sciences 180 (2010) 209–v224.

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

[14] J. Derrac, S. García, F. Herrera, IFS-CoCo: Instance and feature selection based on cooperative coevolution with nearest neighbor rule, Pattern Recognition 43 (2010) 2082–2105.

[15] J. Derrac, S. García, F. Herrera, A survey on evolutionary instance selection and generation, International Journal of Applied Metaheuristic Computing 1 (2010) 60–92.

[16] J. Derrac, S. García, D. Molina, F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, Swarm and Evolutionary Computation 1 (2011) 3–18.

[17] A.E. Eiben, J.E. Smith, Introduction to evolutionary computing, Natural Computing, Springer-Verlag, 2003.

[18] A. Frank, A. Asuncion, UCI machine learning repository, 2010.

[19] A.A. Freitas, Data Mining and Knowledge Discovery with Evolutionary Algorithms, Springer-Verlag, 2002.

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

[21] S. García, J. Derrac, J.R. Cano, F. Herrera, Prototype selection for nearest neighbor classification: Taxonomy and empirical study, IEEE Transactions on Pattern Analysis and Machine Intelligence, in press, doi:10.1109/TPAMI.2011.142.

[22] S. García, A. Fernández, J. Luengo, F. Herrera, Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power, Information Sciences 180 (2010) 2044–2064.

[23] 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.

[24] N. García-Pedrajas, J.A.R. del Castillo, D. Ortiz-Boyer, A cooperative coevolutionary algorithm for instance selection for instance-based learning, Machine Learning 78 (2010) 381–420.

[25] A. Ghosh, L.C. Jain (Eds.), Evolutionary Computation in Data Mining, Springer-Verlag, 2005.

[26] R. Gil-Pita, X. Yao, Evolving edited k-nearest neighbor classifiers, International Journal of Neural Systems 18 (2008) 1–9.

[27] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, Journal of Machine Learning Research 3 (2003) 1157–1182.

[28] I. Guyon, S. Gunn, M. Nikravesh, L.A. Zadeh (Eds.), Feature Extraction: Foundations and Applications, Springer, 2006.

[29] S.Y. Ho, C.C. Liu, S. Liu, Design of an optimal nearest neighbor classifier using an intelligent genetic algorithm, Pattern Recognition Letters 23 (2002) 1495–1503.

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

[31] I. Inza, P. Larrañaga, B. Sierra, Feature subset selection by bayesian networks: a comparison with genetic and sequential algorithms, International Journal of Approximate Reasoning 27 (2001) 143–164.

[32] H. Ishibuchi, T. Nakashima, Evolution of reference sets in nearest neighbor classification, in: Second Asia-Pacific Conference on Simulated Evolution and Learning on Simulated Evolution and Learning (SEAL'98), vol. 1585, Lecture Notes in Computer Science, 1998, pp. 82–89.

[33] M.Z. Jahromi, E. Parvinnia, R. John, A method of learning weighted similarity function to improve the performance of nearest neighbor, Information Sciences 179 (2009) 2964–2973.

[34] R. Jensen, C. Cornelis, Fuzzy-rough instance selection, in: Proceedings of the WCCI 2010 IEEE World Congress on Computational Intelligence, IEEE Congress on Fuzzy Logic, Barcelona Spain, 2010, pp. 1776–1782.

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

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

[37] K. Kim, Artificial neural networks with evolutionary instance selection for financial forecasting, Expert Systems with Applications 30 (2006) 519–526.

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

[39] H. Liu, F. Hussain, C.L. Tan, M. Dash, Discretization: An enabling technique, Data Mining and Knowledge Discovery 6 (2002) 393–423.

[40] H. Liu, H. Motoda (Eds.), Feature selection for knowledge discovery and data mining, The Springer International Series in Engineering and Computer Science, Springer, 1998.

[41] H. Liu, H. Motoda (Eds.), Instance selection and construction for data mining, The Springer International Series in Engineering and Computer Science, Springer, 2001.

[42] H. Liu, H. Motoda, On issues of instance selection, Data Mining and Knowledge Discovery 6 (2002) 115–130.

[43] H. Liu, H. Motoda (Eds.), Computational methods of feature selection, Chapman & Hall/Crc Data Mining and Knowledge Discovery Series, Chapman & Hall/Crc, 2007.

[44] H. Liu, L. Yu, Toward integrating feature selection algorithms for classification and clustering, IEEE Transactions on Knowledge and Data Engineering 17 (2005) 1–12.

[45] E. Mjolsness, D. DeCoste, Machine learning for science: State of the art and future prospects, Science 293 (2001) 2051–2055.

[46] H.S. Nguyen, A. Skowron, J. Stepaniuk, Granular computing: A rough set approach, Computational Intelligence 17 (2001) 514–544.

[47] I.S. Oh, J.S. Lee, B.R. Moon, Hybrid genetic algorithms for feature selection, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (2004) 1424–1437.

[48] A.N. Papadopoulos, Y. Manolopoulos, Nearest Neighbor Search: A Database Perspective, Springer Verlag Telos, 2004.

[49] G.L. Pappa, A.A. Freitas, Automating the design of data mining algorithms: an evolutionary computation approach, Natural Computing, Springer, 2009.

[50] Z. Pawlak, Rough sets, International Journal of Computer and Information Sciences 11 (1982) 341v–356.

[51] Z. Pawlak, Rough Sets: Theoretical Aspects of Reasoning About Data, Kluwer Academic Publishing, 1991.

[52] Z. Pawlak, A. Skowron, Rough sets: some extensions, Information Sciences 177 (2007) 28–40.

[53] Z. Pawlak, A. Skowron, Rudiments of rough sets, Information Sciences 177 (2007) 3–27.

[54] D. Pyle, Data preparation for data mining, The Morgan Kaufmann Series in Data Management Systems, Morgan Kaufman, 1999.

[55] G.J.R. Kohavi, Wrappers for feature selection, Artificial Intelligence 97 (1997) 273–324.

[56] A. Radzikowska, E. Kerre, A comparative study of fuzzy rough sets, Fuzzy Sets and Systems 126 (2002) 137–156.

[57] Y. Saeys, I. Inza, P. Larrañaga, A review of feature selection techniques in bioinformatics, Bioinformatics 19 (2007) 2507–2517.

[58] G. Shakhnarovich, T. Darrell, P. Indyk (Eds.), Nearest-Neighbor Methods in Learning and Vision: Theory and Practice, The MIT Press, 2006.

[59] D.J. Sheskin, Handbook of Parametric and Nonparametric Statistical Procedures, fifth ed., Chapman & Hall/CRC, 2011.

[60] A. Skowron, J. Stepaniuk, Tolerance approximation spaces, Fundamenta Informaticae 27 (1996).

[61] A. Skowron, J. Stepaniuk, R.W. Swiniarski, Approximation spaces in rough-granular computing, Fundamenta Informaticae 100 (2010).

[62] R. Slowinski, D. Vanderpooten, A generalized definition of rough approximations based on similarity, IEEE Transactions on Knowledge and Data Engineering 12 (2000).

[63] C. Stanfill, D. Waltz, Toward memory-based reasoning, Communications of the ACM 29 (1986) 1213–1228.

[64] M. Stone, Cross-validatory choice and assessment of statistical predictions (with discussion), Journal of the Royal Statistical Society B 36 (1974) 111v–147.

[65] I. Triguero, J. Derrac, S. García, F. Herrera, A taxonomy and experimental study on prototype generation for nearest neighbor classification, IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, in press, doi: 10.1109/TSMCC.2010.2103939.
[66] I. Triguero, S. García, F. Herrera, IPADE: Iterative prototype adjustment for nearest neighbor classification, IEEE Transactions on Neural Networks 21 (2010) 1984–1990.
[67] E. Tsang, D. Chen, D. Yeung, X. Wang, J.T. Lee, Attributes reduction using fuzzy rough sets, IEEE Transactions on Fuzzy Systems 16 (2008) 1130–1141.
[68] K. Weinberger, L. Saul, Distance metric learning for large margin nearest neighbor classification, Journal of Machine Learning Research 10 (2009) 207–244.
[69] F. Wilcoxon, Individual comparisons by ranking methods, Biometrics Bulletin 1 (1945) 80–83.
[70] D. Wilson, T. Martinez, Improved heterogeneous distance functions, Journal of Artificial Intelligence Research 6 (1997) 1–34.
[71] D.R. Wilson, T.R. Martinez, Reduction techniques for instance-based learning algorithms, Machine Learning 38 (2000) 257–286.
[72] I.H. Witten, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques, second ed., Morgan Kaufmann Series in Data Management Systems, Morgan Kaufmann, 2005.
[73] X. Wu, V. Kumar (Eds.), The top ten algorithms in data mining, Data Mining and Knowledge Discovery, Chapman & Hall/CRC, 2009.
[74] X. Yang, J. Yang, C. Wu, D. Yu, Dominance-based rough set approach and knowledge reductions in incomplete ordered information system, Information Sciences 178 (2008) 1219–1234.
[75] L.A. Zadeh, Fuzzy sets, Information and Control 8 (1965) 338–353.

3.2 On the Use of Evolutionary Feature Selection for Improving Fuzzy Rough Set Based Prototype Selection

- J. Derrac, N. Verbiest, S. García, C. Cornelis, F. Herrera, On the Use of Evolutionary Feature Selection for Improving Fuzzy Rough Set Based Prototype Selection. Soft Computing 17:2 (2013) 223-238, doi: 10.1007/s00500-012-0888-3
  - Status: **Published**.
  - Impact Factor (JCR 2011): 1.880.
  - Subject Category: Computer Science, Artificial Intelligence. Ranking 30 / 111 (**Q2**).
  - Subject Category: Computer Science, Interdisciplinary Applications. Ranking 24 / 99 (**Q1**).

Soft Comput (2013) 17:223–238
DOI 10.1007/s00500-012-0888-3

FOCUS

On the use of evolutionary feature selection for improving fuzzy rough set based prototype selection

J. Derrac · N. Verbiest · S. García ·
C. Cornelis · F. Herrera

Published online: 20 July 2012

Abstract The *k*-nearest neighbors classifier is a widely used classification method that has proven to be very effective in supervised learning tasks. In this paper, a fuzzy rough set method for prototype selection, focused on optimizing the behavior of this classifier, is presented. The hybridization with an evolutionary feature selection method is considered to further improve its performance, obtaining a competent data reduction algorithm for the 1-nearest neighbors classifier. This hybridization is performed in the training phase, by using the solution of each preprocessing technique as the starting condition of the other one, within a cycle. The results of the experimental study, which have been contrasted through nonparametric statistical tests, show that the new hybrid approach obtains very promising results with respect to classification accuracy and reduction of the size of the training set.

Keywords Prototype selection · Feature selection · Data reduction · Fuzzy rough sets · Evolutionary algorithms · Nearest neighbor

J. Derrac (✉) · C. Cornelis · F. Herrera
Department of Computer Science and Artificial Intelligence, CITIC-UGR (Research Center on Information and Communications Technology), University of Granada, 18071 Granada, Spain
e-mail: jderrac@decsai.ugr.es

N. Verbiest · C. Cornelis
Department of Applied Mathematics and Computer Science, Ghent University, Ghent, Belgium

S. García
Department of Computer Science, University of Jaén, 23071 Jaén, Spain

1 Introduction

Supervised classification is one of the most useful techniques in machine learning (Mjolsness and DeCoste 2001; Alpaydin 2010; Witten et al. 2011). Categorizing new objects using data stored in a given training set has become a critical task in many real-world applications of data mining and pattern recognition.

The *k*-nearest neighbors classifier (*k*-NN) (Cover and Hart 1967; Shakhnarovich et al. 2006) is one of the most relevant algorithms in data mining (Wu and Kumar 2009). It is a nonparametric classifier which simply uses the entire input data set to establish the classification rule. Thus, the effectiveness of the classification process performed by *k*-NN relies mainly on the quality of the training data (Aha et al. 1991). Furthermore, it is important to note that its main drawback is its relative inefficiency as the size of the problem increases, regarding both the number of instances in the data set and the number of features which will be used in the computation of the similarity (distance) function (Chen et al. 2009; Weinberger and Saul 2009).

However, the overwhelming amount of data available nowadays in any field of research (Bell et al. 2009) poses new problems when using the *k*-NN classifier. Gathering, understanding and processing such data often requires the use of advanced tools for managing the represented knowledge in a suitable way. In this sense, many approaches have been proposed to improve the performance of *k*-NN (Triguero et al. 2010; Destercke 2012). Some of the most effective ones work directly over the training data, instead of modifying the computation of the *k*-NN rule. They preprocess the initially available data, aiming to improve the algorithms in terms of efficiency and efficacy.

Data reduction (Pyle 1999) is a data preprocessing task whose main objective is to reduce the original training set.

By removing noisy and irrelevant data—harmful for the majority of machine learning methods—data reduction can help to avoid excessive storage and time requirements, easing and enabling machine learning techniques to deal with large data sets. The best known data reduction processes are feature selection (FS) (Liu and Motoda 2007), feature generation/extraction (Guyon et al. 2006), attribute discretization (García et al. 2012b), prototype generation (Triguero et al. 2012) and prototype selection (PS) (García et al. 2012a).

One of the most successful families of data reduction methods has been originated by evolutionary computation (Eiben and Smith 2003; Ghosh and Jain 2005). Evolutionary algorithms, search algorithms inspired by natural populations to evolve solutions, have been applied to different data reduction problems, modeling them as combinatorial problems (Freitas 2002; Cano et al. 2003; Pappa and Freitas 2009). A remarkable number of evolutionary data reduction techniques have been focused on optimizing the *k*-NN rule.

Fuzzy sets (Zadeh 1965) and rough sets (Pawlak 1982; Pawlak and Skowron 2007b) address two important, complementary characteristics of imperfect data and knowledge: the former model vague information by expressing that objects belong to a set or relation to a given degree, while the latter provide approximations of concepts in the presence of incomplete information (Kusunoki and Inuiguchi 2010). A hybrid fuzzy rough set model was first proposed in (Dubois and Prade 1990), later extended and/or modified by many authors, being applied successfully in various domains (Radzikowska and Kerre 2002; De Cock et al. 2007; Tsang et al. 2008; He and Wu 2011; Zhai 2011). Possibly, the most notable capability of these extensions is that they enable practitioners to apply rough sets analysis directly over data sets representing continuous data, in contrast with pure rough sets methods, which cannot be applied over continuous data sets without discretizing them at a previous step.

In this paper, a hybrid model for data reduction combining fuzzy rough sets and evolutionary algorithms is proposed. A PS algorithm is developed by estimating the quality of every training instance through a fuzzy rough set based quality measure. The solutions obtained by this method are used to adjust the search of an evolutionary FS algorithm. Both the PS and the FS algorithms are applied in succession, using the 1-NN classifier as a wrapper for evaluating the solutions. At the end, the subsets of features and prototypes selected are gathered to generate a final data set, which is used in the classification phase as reference.

We have tested our approach (which we have termed EFS-RPS, evolutionary feature selection for fuzzy rough set based prototype selection) in a wide selection of supervised classification domains, considering 38 different problems. The results have been contrasted by using several non-parametric statistical tests (Sheskin 2011), reinforcing the conclusions obtained in the experiments.

The rest of the paper is organized as follows: Sect. 2 shows an introduction to the main topics of the study. Section 3 describes EFS-RPS and its main characteristics. Section 4 details the experimental study performed to test the performance of the new technique. Finally, Sect. 5 sums up our main conclusions.

2 Data reduction preliminaries

This section gives some preliminaries on data reduction techniques, fuzzy rough sets and evolutionary algorithms:

- Section 2.1 describes feature selection and its application for enhancing the *k*-NN rule.
- Section 2.2 surveys prototype selection and some notable characteristics of the field.
- Section 2.3 recalls some definitions of fuzzy rough sets and various works related to data reduction.
- Section 2.4 reviews the use of evolutionary algorithms for the reduction of training sets in *k*-NN based classifiers.

Throughout the section, the following definitions will be used:

- The data set X consists of N instances which are defined by a set A of M attributes (features) in an M-dimensional space and a class (decision) attribute c. Attribute values in M should be normalized in the interval [0, 1].
- Each instance X_p is defined by $X_p = (X_{p1}, X_{p2}, ..., X_{pM}, X_{pc})$, where X_{pi} is the value of the i-th feature of the p-th instance. The class attribute of the instance is determined by X_{pc}, which means that X_p belongs to the class c.
- The data set is split into two different subsets: A training set TR and a test set TS. After the application of a data reduction algorithm (a PS or a FS one) over TR, a reference set RS $\subseteq$ TR is obtained.

2.1 Feature selection

One of the main data reduction techniques is FS. Its goal is to select the most appropriate subset of features from the initial data set. It aims to eliminate irrelevant and redundant features to obtain a simple and accurate classification system (Liu and Motoda 2007).

FS can be defined as follows: Given a data set composed by TR and TS, a FS algorithm searches for a subset of features $B \subseteq A$. The RS set is built from TR, considering only the features selected in B. Instances from TS are then classified by a data mining algorithm using RS as reference.

There are three main categories into 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 (Kohavi and John 1997).
- *Filtering methods*, where the selection criterion is independent of the learning algorithm. In these methods, the selection is guided by data related measures (for example, separability measures) (Guyon and Elisseeff 2003).
- *Embedded methods*, where the search for an optimal subset of features is performed within the classifier construction (Saeys et al. 2007).

The most popular algorithms for FS are the classical sequential ones. Forward sequential and backward sequential selection (Liu and Motoda 1998) are the best-known ones. They begin with a feature subset and sequentially add or remove features if they improve the quality of the selection until the algorithm finishes. Other remarkable FS methods are FOCUS (Almuallim and Dietterich 1991) and the RELIEF family (Kira and Rendell 1992).

Despite the popularity of these classical methods, many other approaches based on heuristic search can be found in the literature (Stracuzzi and Utgoff 2004; Shie and Chen 2008). Complete surveys, analyzing both classical and advanced approaches to FS, can be found in the literature (Guyon and Elisseeff 2003; Liu and Yu 2005; Saeys et al. 2007).

2.2 Prototype selection

PS methods are data reduction methods whose objective is to isolate the smallest set of instances which enable the k-NN classifier to predict the class of a query instance with the same quality as the initial data set (Liu and Motoda 2001).

PS can be defined as follows: Given a data set composed by TR and TS, a PS algorithm obtains prototypes as a subset of instances RS $\subseteq$ TR. Instances from TS are then classified by the k-NN classifier using RS as reference.

Depending on the strategy followed, PS methods can be categorized into three classes: *preservation methods*, which aim to obtain a consistent subset from the training data, ignoring the presence of noise; *noise removal methods*, which aim to remove noise both in the boundary points (instances near to the decision boundaries) and in the inner points (instances far from the decision boundaries), and *hybrid methods*, which perform both objectives simultaneously (García et al. 2012a).

PS methods are sometimes dependent on the k value set on the definition of the k-NN classifier. In (Wilson and Martinez 2000), it is stated that setting $k > 1$ decreases the sensitivity of the algorithm to noise and tends to smooth the decision boundaries. In some PS algorithms, a value $k > 1$ may be convenient, when the interest lies in protecting the classification task against noisy instances. Therefore, they state that it may be appropriate to find a value of k to use in the reduction process, and then recompute the best value of k in the classification phase. In this work, we have employed the value $k = 1$, to give the classifier the greatest possible sensitivity to noise during the reduction process. In this manner, an evolutionary PS algorithm can detect better the noisy instances and the redundant ones present in the training set.

Despite the variety of PS methods developed in the last decades (with some remarkable proposals such as CNN (Hart 1968), ENN (Wilson 1972) or the IB (Aha et al. 1991) and DROP (Wilson and Martinez 2000) families), improvements in storage reduction, noise tolerance, generalization accuracy and time requirements are reported still nowadays, with the development of new PS methods (García et al. 2008; Ferrandiz and Boullé 2010; Franco et al. 2010; Quirino et al. 2010). They have become a proof of the topical nature of this field, which continues to attract the interest of many research communities in the search for new ways to further improve the performance of the k-NN classifier. More information about the PS field can be found at the SCI2S thematic public website on *Prototype Reduction in Nearest Neighbor Classification.*[1]

2.3 Fuzzy rough sets for data reduction

Rough set theory (Pawlak 1991; Pawlak and Skowron 2007a) provides a methodology for data analysis based on the approximation of concepts in a decision system $(X, A \cup \{c\})$, in which X is a set of instances, A is a set of conditional attributes and c is the decision or class attribute.

The theory revolves around the notion of (in)discernibility: the ability to distinguish between instances, based on their attribute values. When fuzzy rough sets are used, indiscernibility is typically modeled by means of a fuzzy tolerance relation R in X. In this paper, R is defined as, for X_x and X_y in $X (x, y \in \{1, \ldots, n\})$,

$$R(X_x, X_y) = \mathcal{T}(\underbrace{R_a(X_x, X_y)}_{a \in A}) \tag{1}$$

where $\mathcal{T}$ is a t-norm, which is an associative, commutative mapping $\mathcal{T} : [0,1]^2 \to [0,1]$, increasing in both arguments and such that $\forall s \in [0,1] : \mathcal{T}(s,1) = s$. In this paper we will use the Łukasiewicz t-norm, defined as follows: $\mathcal{T}(s,t) = \max(0, s+t-1)$ for $s,t \in [0,1]$. Note that, as a t-norm is associative and commutative, it can be extended unambiguously for M arguments as in Eq. (1).

The indiscernibility for one attribute $R_a(X_x, X_y)$ is given by

[1] http://sci2s.ugr.es/pr/.

$$R_a(X_x, X_y) = 1 - (X_{xa} - X_{ya})^2 \tag{2}$$

if a is a quantitative (real) attribute, and

$$R_a(X_x, X_y) = \begin{cases} 1 & \text{if } X_{xa} = X_{ya} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

when a is nominal (discrete).

Given R, the fuzzy-rough positive region of X is defined as the fuzzy set $\mathrm{POS_A}$ in X

$$\mathrm{POS_A}(X_x) = \min_{X_y \in X} \mathscr{I}(R(X_x, X_y), R_c(X_x, X_y)) \tag{4}$$

where $\mathscr{I}$ represents an implicator, which is a mapping $\mathscr{I} : [0,1]^2 \to [0,1]$, decreasing in the first and increasing in the second argument, and for which $\mathscr{I}(0,0) = 1, \mathscr{I}(1,1) = 1, \mathscr{T}(0,1) = 1$ and $\mathscr{I}(1,0) = 0$. In this paper we will use the Łukasiewicz implicator, defined as follows: $\mathscr{I}(s,t) = \min(1, 1 - s + t)$ for s, t $\in [0, 1]$. The indiscernibility w.r.t. the decision class is defined as follows:

$$R_c(X_x, X_y) = \begin{cases} 1 & \text{if } X_{xc} = X_{yc} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

The idea of the fuzzy-rough positive region is that instances on the border of a class (that is, for which there exists a similar instance in another class) will have a small membership value to $\mathrm{POS_A}$ compared to instances in the center of a class. This makes the fuzzy-rough positive region suitable to measure the quality of an instance as a typical representative of its class. Most applications of fuzzy rough sets for FS and PS data reduction are based on this approach (Jensen and Shen 2007, 2009; Cornelis et al. 2010; Jensen and Cornelis 2010).

2.4 Evolutionary algorithms for data reduction

Recently, the use of evolutionary algorithms in data reduction problems has become common in the machine learning field. This subsection surveys some interesting approaches for evolutionary FS and evolutionary PS.

In (Cano et al. 2003), a complete study on the use of evolutionary algorithms for prototype selection is carried out, highlighting four evolutionary methods to complete this task: CHC adaptive search algorithm (Eshelman 1991), steady-state genetic algorithm (SSGA) (Whitley 1989), generational genetic algorithm and population-based incremental learning. They concluded that the evolutionary algorithms selected outperform classical algorithms both in reduction rates and classification accuracy.

Other interesting evolutionary proposals for PS can be found in (García et al. 2008; Gil-Pita and Yao 2008; Ishibuchi and Nakashima 1998; Kuncheva 1995; Derrac et al. 2010a; García-Pedrajas et al. 2010). For a detailed survey on the field see (Derrac et al. 2010b).

Regarding FS, most of the evolutionary approaches are based on genetic algorithms, using both filter and wrapper approaches (Casillas et al. 2001; Gonzalez and Perez 2001; Oh et al. 2004; Rokach 2008). Another interesting proposal is (Inza et al. 2001), where an estimation of distribution algorithm based on bayesian networks is presented.

It is also possible to find evolutionary applications of simultaneous PS and FS. Both (Kuncheva and Jain 1999) and (Ishibuchi et al. 2001) propose a genetic algorithm to perform simultaneously the editing of the instance set and selection of the feature set. Another popular dual method is IGA (Ho et al. 2002), an intelligent genetic algorithm designed to tackle both PS and FS problems simultaneously, by the introduction of a special orthogonal crossover operator.

3 EFS-RPS: evolutionary feature selection for fuzzy rough set based prototype selection

In this section we describe the main components of EFS-RPS and its implementation details. The organization of this section follows a bottom-up order in which:

- Firstly, in Sect. 3.1, the PS algorithm based on fuzzy rough sets which forms the core of EFS-RPS is presented.
- Secondly, in Sect. 3.2, we describe the evolutionary algorithm developed for performing the FS process.
- Thirdly, in Sect. 3.3, we detail the way in which the subsets of features and prototypes obtained through the above methods are combined for preprocessing reference sets for the 1-NN classifier.
- Finally, in Sect. 4.4, a full description on the EFS-RPS is given, as a combination of all the components described before.

```
   Input: The full set of instances $TR$
   Output: A subset of prototypes $RS$
 1 $CP = TR$ (Current Prototypes);
 2 Compute $POS_A(x)$ for each $x \in CP$;
 3 Sort prototypes in $CP$ by their $POS_A(x)$ value in increasing
   order;
 4 Compute nearest neighbors of $TR$ in $CP$ ;
 5 bestAcc = 1-NNlooAccuracy($CP$);
 6 $RS = CP$;
 7 continue = true ;
 8 while $CP$ not empty do
 9     Mark the next prototype of RS, $y$, for deletion:
       $CP = CP - \{y\}$;
10     Update nearest neighbors of $TR$ in $CP$ ;
11     newAcc = 1-NNlooAccuracy($CP$);
12     if $newAcc > bestAcc$ then
13         $RS = CP$;
14         bestAcc = newAcc;
15     end
16 end
```

Algorithm 1 A PS algorithm based on fuzzy rough set theory

3.1 A fuzzy rough set PS procedure

The quality of the instances in the training set can be assessed following the concepts of fuzzy rough set theory defined before and the definition of the fuzzy positive region of an instance $x \in X$. The membership of an instance to the positive region with respect to the current subset of features considered $B \subseteq M$ can serve as a noise measure for it.

In this fuzzy rough set PS procedure, instances are ordered with respect to their fuzzy positive region, in increasing order. Then, noisy instances are pruned iteratively, and the subsequently found prototype subsets are evaluated with respect to classification accuracy (accuracy is estimated by using a leave-one-out procedure with a 1-NN classifier, which we denote by 1-NNlooAccuracy). Instances with the same fuzzy positive region value are analyzed simultaneously, that is, marked for deletion at the same time. The best subset found is then identified as the subset of prototypes RS, which is the output of the algorithm. Algorithm 3 shows the pseudocode of this procedure.

Through the procedure, a list with the nearest neighbor of each prototype in TR is maintained. These nearest neighbors can only belong to the set RS; hence, every time a prototype from RS is removed, the list of neighbors is updated and the neighbor of every prototype of TR which is now missing is recomputed.

This neighbors list is used for estimating the 1-NN leave-one-out accuracy of the current RS set selected [through the 1-NNlooAccuracy (RS) procedure]. It helps the PS method to avoid the necessity of recomputing the nearest neighbors of each prototype in TR every time the procedure is used, thus saving computational resources.

Thanks to this optimization, the cost of the PS procedure can be computed by using the concept of partial evaluations. Throughout the FS and PS process of EFS-RPS, the computational resources spent are registered in the form of solutions evaluations. Every time a full classification of the TR is performed to estimate the accuracy of a solution, a full evaluation is spent.

In the specific case of this PS procedure, the complete cost is defined as follows:

- A full evaluation is spent the first time the nearest neighbor list is computed.
- Every time a neighbor has to be updated (because the old neighbor has been removed from RS) a partial evaluation is spent:

$$partialEvaluation = \frac{1}{\#\text{instances in TR}} \tag{6}$$

Therefore, the total cost of the PS algorithm can be redefined as

$$PSCost = 1 + \frac{\#\text{neighborsupdated}}{\#\text{instances in TR}} \tag{7}$$

This partial evaluations procedure, inspired by the one developed in (García et al. 2008), allows us to define a fair computational cost measure for the PS algorithm, which correctly represents the savings obtained through the use of the list of neighbors.

3.2 Searching features using an evolutionary algorithm

The second key element of EFS-RPS is its search method for selecting subsets of features. To accomplish this task, we have chosen SSGA as the evolutionary algorithm to perform the search.

A SSGA is a genetic algorithm in which only a reduced set of offspring is produced in each generation (two, in most cases). Parents are chosen to produce offspring and then a decision is made as to which individuals in the population will be selected for deletion in order to make room for the new offspring. Algorithm 2 shows the pseudocode of SSGA.

```
  Input: A population
  Output: An optimized population
1 Initialize population;
2 while Termination criterion not satisfied do
3     Select two parents from the population;
4     Create two offspring using crossover and mutation;
5     Evaluate the offspring with the fitness function;
6     Select two individuals in the population, which may be
      replaced by the offspring;
7     Decide if this/these individuals will be replaced;
8 end
```

Algorithm 2 SSGA pseudocode

The fitness function of our SSGA pursues a dual objective: The main task of the method is to search for subsets of features which increase the accuracy of the 1-NN classifier. However, a second task should be to reduce the size of the subsets selected, if this does not harm the accuracy rates obtained.

Hence, following the same set-up as in (Cano et al. 2003), where a similar approach is used in the core of evolutionary PS methods, for a given solution J (chromosome) of the SSGA, we define two variables:

- *AccRate*: The classification accuracy of a 1-NN classifier (1-NNAccuracy) when classifying the full training set using only the currently selected subset of features as a reference (and leave-one-out as validation scheme).

$$AccRate(J) = \text{1-NNAccuracy}(J) \tag{8}$$

- *RedRate* The rate of reduction achieved over the currently selected (maintained) features.

$$RedRate(J) = \frac{1.0 - \text{\#FeaturesSelected(J)}}{M} \tag{9}$$

Both variables are adjusted through a real-valued weighting factor α to equalize the strength of each term in the resulting fitness value. The final fitness function of the SSGA can be defined as

$$Fitness(J) = \alpha \cdot AccRate(J) + (1 - \alpha) \cdot RedRate(J) \tag{10}$$

The α value should be kept very high ($\alpha = 0.99$ turned out to be the best choice in our preliminary experiments) in order to avoid those cases in which an excessive deletion of features could be favored too much by the fitness function, resulting in a selection of an insufficient number of features for the final classification stage.

The configuration details of the SSGA are as follows:

- *Codification* The SSGA will use binary chromosomes to represent the solutions. Each bit will represent the state of each feature in the training set (**1** if the feature is selected; **0** if it is deleted).
- *Crossover operator* A two-point crossover operator has been considered. In each generation, this operator is applied twice, obtaining two offspring.
- *Mutation operator* The bit-flip mutation operator (changing the value of the selected allele from 0 to 1, and vice versa) is applied to each offspring produced, with a given probability per bit.
- *Selection of parents* A binary tournament procedure will be used to select parents in each generation.
- *Replacement strategy* The two worst individuals of the population are chosen for replacement, only if their fitness value is lower than the offspring's.

3.3 Simultaneous assessment of features and prototypes through the 1-NN classifier

While the search process performed by EFS-RPS is carried out, we will need to assess the quality of the solutions obtained. Mostly, this operation will consist of gathering two solutions (one representing a prototype subset, and another one representing a subset of features), combining them and estimating their quality through the 1-NN classifier.

Once both solutions have been gathered, their assessment is carried out by performing the following steps:

1. A copy of the training set is obtained and pruned, keeping only those prototypes indicated by the PS solution.
2. After the prototypes have been isolated, their features are also pruned, keeping this time only those indicated by the FS solution. The resulting subset is identified as the reference subset.
3. The 1-NN classifier is used to classify all the original training instances. This 1-NN classifier will use as reference set only the data preprocessed in the previous step.
4. The accuracy of this classification (that is, the ratio of training instances correctly classified over the total number of training instances), a value in [0, 1], will be used as the quality of the solutions.

This simple method allows us to evaluate subsets of features and prototypes within the general EFS-RPS framework. Since it involves the classification of the full training set, performing it has a cost associated of a full evaluation.

3.4 EFS-RPS global model

The EFS-RPS is composed by the fuzzy rough set based PS method and the SSGA FS algorithm defined before. Through the procedure for evaluating simultaneously feature and prototype subsets (defined in the previous subsection), EFS-RPS can merge the two search processes in an effective way, enabling the framework to obtain improved results from the existing synergy between the two basic data reduction methods on which it is based.

EFS-RPS begins by selecting a candidate set of prototypes, *bestPS*. This subset of the training set is used as reference for the SSGA, during a fixed number of evaluations (*Cycle Length*). Every time this limit is reached, the candidate set of prototypes is recomputed through the PS method, but considering only the best subset of features found so far, *bestFS*. If the new subset of prototypes computed is better than the previous one (in terms of leave-one-out accuracy), the latter is updated.

These processes are repeated until the algorithm is close to its end (at that point, no further update of the *bestPS* subset is allowed). Once the *Evaluations limit* is reached, the best subsets of prototypes and features found so far, *bestPS* and *bestFS* are used to prepare the final reference set, RS, which is obtained as the output of the algorithm. Algorithm 4 shows the EFS-RPS pseudocode.

```
Input: A training set $TR$, Evaluations limit, Cycle Length
Output: A reduced reference set RS
1  $bestFS$=$M$;
2  ($bestAcc$,$bestPS$)=PS-procedure($TR$,$bestFS$);
3  Initialize population of SSGA;
4  optimizePS=true;
5  cycle=0;
6  while Evaluations limit not reached do
7      $bestFS$=FSprocedureGeneration();
8      if evaluations spent > ($\gamma \cdot$ Evaluations limit) then
9          optimizePS=false;
10     end
11     if optimizePS AND cycle=Cycle Length then
12         //Current best features found are used in the
           evaluation of the solutions of the PS method;
13         ($newAcc$,$newPS$)=PSprocedure($TR$,$bestFS$);
14         if $newAcc$ > $bestAcc$ then
15             $bestAcc$=$newAcc$;
16             $bestPS$=$newPS$;
17         end
18         cycle=0;
19     end
20     cycle+=2;
21 end
22 $RS$=reduceSet($TR$,$bestPS$,$bestFS$);
```

Algorithm 3 The global EFS-RPS model

The main steps of the algorithm are detailed as follows:

- Instructions 1–2 extract a preliminary set of prototypes by applying the fuzzy rough PS procedure detailed in Sect. 3.1. In this first application of the PS procedure, all the features of the problem are considered.
- Instruction 3 initializes the population of the SSGA and its chromosomes are evaluated (using the procedure detailed in Sect. 3.3. The prototypes considered in this step are those stored in *bestPS*. Instructions 4 and 5 initialize *optimizePS* and *cycle* variables.
- Instruction 7 performs a full generation of the SSGA (again, the new chromosomes are evaluated using the procedure detailed in Sect. 3.3).
- Instructions 8 and 9 check if the end of the algorithm is close; that is, when more than $\gamma \cdot$ *Evaluations limit* evaluations have been spent (γ should be set close to 1, for example, $\gamma = 0.75$). If that limit is reached, the set of prototypes *bestPS* is no further optimized (see Instructions 11–20). Disabling these instructions in the last generations of the algorithms will help to improve the convergence capabilities of the SSGA.
- Instruction 10 checks if the set of prototypes *bestPS* has to be improved. If this phase is enabled (see Instructions 8 and 9) and *Cycle Length* evaluations have been spent since the last time this phase was carried out, the *bestPS* set can be improved.
- Instruction 13 generates a new candidate set of prototypes, *newPS* (only the features of the best subset found so far by the SSGA are considered in this case). The accuracy obtained by using this new set of prototypes is computed, *newAcc*. If it is higher than *bestAcc*, then the set *bestPS* is updated (Instructions 14–17).
- Instruction 22 gathers the best solutions found by the SSGA and the PS procedure, and creates a final reference set RS with those prototypes and features selected in the solutions.

The EFS-RPS algorithm loop is carried out until the specified limit of evaluations is reached. Then, the RS subset generated can be used as a reference set for the 1-NN classifier to classify new test instances.

4 Experimental study

This section describes the experimental study performed to test the performance of EFS-RPS:

- Section 4.1 lists the supervised classification problems considered and their main characteristics.
- Section 4.2 provides a description of the algorithms considered in the comparison and a definition of their parameter values.
- Section 4.3 describes the nonparametric statistical procedures considered for contrasting the results of the study.
- Section 4.4 shows the results obtained and analyzes them.

4.1 Data sets

We have selected a set of 38 classification data sets for our experimental study. These are well-known problems in the area, taken from the KEEL-dataset repository (Alcalá-Fdez et al. 2008, 2011)[2] and the UCI repository (Frank and Asuncion 2010). Table 1 summarizes their main characteristics. For each data set, we provide its number of instances (#Ins.), attributes (#At.) and classes (#Cl.).

The data sets considered are partitioned by using the 10-fold cross-validation (10-fcv) procedure (enabling us to follow a 5x10-fold cross-validation set-up in the study), and their values are normalized in the interval [0, 1] to equalize the influence of attributes with different range domains. In addition, instances with missing values have been discarded before the execution of the methods over the data sets.

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

Table 1 Description of the 38 data sets used in the study

Data set	#Ins.	#At.	#Cl.	Data set	#Ins.	#At.	#Cl.
Australian	690	14	2	Housevotes	435	16	2
Automobile	205	25	6	Iris	150	4	3
Balance	625	4	3	Led7Digit	500	7	10
Bands	539	19	2	Lymphography	148	18	4
Breast	286	9	2	Mammographic	961	5	2
Bupa	345	6	2	Monks	432	6	2
Car	1,728	6	4	New thyroid	215	5	3
Cleveland	303	13	5	Pima	768	8	2
Contraceptive	1,473	9	3	Saheart	462	9	2
Crx	690	15	2	Sonar	208	60	2
Dermatology	366	34	6	Spectfheart	267	44	2
Ecoli	336	7	8	Tae	151	5	3
Flare-solar	1,066	9	2	Tic-tac-toe	958	9	2
German	1,000	20	2	Vehicle	946	18	4
Glass	214	9	7	Vowel	990	13	11
Haberman	306	3	2	Wine	178	13	3
Hayes-Roth	160	4	3	Wisconsin	699	9	2
Heart	270	13	2	Yeast	1,484	8	10
Hepatitis	155	19	2	Zoo	101	16	7

4.2 Algorithms and parameter settings

In order to show the capabilities of EFS-RPS as a data preprocessor for 1-NN we have selected a representative set of comparison methods, including several evolutionary and fuzzy rough set based ones. The preprocessed training sets obtained as a result of the application of all these methods (including EFS-RPS) will be evaluated using a 1-NN classifier to classify the test (unseen) data. Euclidean distance will be considered in all the methods, whereas the overlap metric is considered for nominal attributes.

The comparison methods selected are the following:

- Evolutionary data reduction methods:
  - *FS-SSGA* A steady-state genetic algorithm for FS. This method follows the same design as the evolutionary component of EFS-RPS, but without including any kind of PS process. Hence, it is only focused on searching the best possible subset of features through a wrapper based evolutionary search.
  - *PS-SSGA* A steady-state genetic algorithm for PS. This method shares the same set-up as FS-SSGA, but it is focused on selecting prototypes, instead of features (it also uses binary chromosomes). Its objective is to find the most representative subset of prototypes from the training set through the evolutionary search process.
  - *FPS-SSGA* A steady-state genetic algorithm for simultaneous FS and PS. This method shares the same set-up as FS-SSGA and PS-SSGA, but both features and prototypes are encoded in the chromosomes. As output, this method will select a subset of features and a subset of prototypes, which are combined in the same way as the solutions of EFS-RPS.
- Fuzzy rough set data reduction methods:
  - *PS-FRW* A fuzzy rough set wrapper algorithm for PS. This method follows the same design as the PS component of EFS-RPS.
  - *FS-RST* A fuzzy rough set based feature selection method. It performs a heuristic search among the features of the training data, choosing the best ones according to how well they represent the full training set (using a measure of discernibility to evaluate the different subsets of features found). More details can be found in (Cornelis et al. 2010).
- Other algorithms:
  - *EIS-RFS* A hybrid data preprocessing method, which incorporates FS-RST and PS-SSGA for performing simultaneous FS and PS. A full description of this method can be found in (Derrac et al. 2012).
  - *1-NN* The 1-NN classifier is also included in the comparison. Its results, unmodified by any data preprocessing method, will give an insight of how well the rest of algorithms are improving the behavior of the base classifier, in terms of accuracy.

Table 2 Parameter specification for the algorithms tested in the experimentation

Algorithm	Parameters
EFS-RPS	Evaluations: 10,000, Population size: 50, Crossover probability: 1.0, Mutation probability: 0.005 per bit, α: 0.99
	Cycle Length: 100, γ: 0.75
PS-SSGA	Evaluations: 10,000, Population size: 50, Crossover probability: 1.0, Mutation probability: 0.005 per bit, α: 0.5
FS-SSGA	Evaluations: 10,000, Population size: 50, Crossover probability: 1.0, Mutation probability: 0.005 per bit, α: 0.99
FPS-SSGA	Evaluations: 10,000, Population size: 50, Crossover probability: 1.0, Mutation probability: 0.005 per bit
	α (instances): 0.5, α (features): 0.99
PS-FRW	–
FS-RST	*MaxGamma*: 1.0
EIS-RFS	Evaluations: 10,000, Population size: 50, Crossover probability: 1.0, Mutation probability: 0.005 per bit, α: 0.5
	MaxGamma: 1.0, UpdateFS: 100, β: 0.75
1-NN	–

Many different configurations can be established for each combination of domain and method. However, for the sake of a fair comparison, we have selected a fixed set of parameters for each method, which will be applied for all the data sets. Table 2 summarizes them.

Essentially, 10,000 evaluations are allowed for every method. SSGA base parameters are set-up to a classical configuration, and the α value in the fitness function is set to 0.5 if the reduction rate is computed over instances [as recommended in Cano et al. (2003)] and to 0.99 if it is computed over features [as recommended in Derrac et al. (2010a)]. The rest of parameters are set to the values recommended by the authors of each technique.

4.3 Statistical procedures

We have considered the use of hypothesis testing techniques to provide statistical support for the analysis of the results of the experimental study. Concretely, we will use nonparametric tests (Sheskin 2011), since the initial conditions that guarantee the reliability of the parametric tests (independence, normality and homocedasticity) may not be satisfied, causing the statistical analysis to lose credibility (García and Herrera 2008; García et al. 2009).

Throughout the study, we perform several multiple comparisons between the algorithms considered. To do so, we will use the Friedman test in order to detect statistical differences among a group of results. A second property of this test is that the ranks computed for obtaining the Friedman statistic can be also considered to sort the algorithm by its relative performance (where the lower the rank obtained, the better the performance of the algorithm). The process followed to compute the final ranks is as follows:

1. Gather observed results for each pair algorithm/data set (for example, average the results obtained after the cross-validation process).
2. For each data set, rank the values from 1 (best result) to n (worst result), where n is the number of algorithms considered in the comparison. If ties appear, assign midranks.
3. Average the ranks obtained in all data sets to obtain the final rank.

After computing the ranks, if the p-value of the Friedman test is significantly low (at a 0.05 level of significance), the existence of significant differences between the algorithms evaluated is assumed. From this point, a control algorithm can be chosen (the one with the lowest rank, that is, the best performing one), and post-hoc procedures (in our case the Holm and Finner procedures (García et al. 2010)) can be applied to determine which algorithms are significantly outperformed by the control one.

More information about these tests and other statistical procedures specifically designed for use in the field of machine learning can be found at the SCI2S thematic public website on *Statistical Inference in Computational Intelligence and Data Mining*.[3]

4.4 Results and analysis

In this subsection we report the results obtained in the full experimental study. Table 3 shows the average accuracy results obtained in the test phase (considering a 5x10-fold cross-validation set-up, that is, averaging the results of five independent schemes of 10-fold cross-validation). For each algorithm and data set, the average accuracy and standard deviation are provided. The best result in each data set is highlighted in bold. Moreover, the table also provides average results over all data sets and the number of times that each algorithm obtains the best result for a single data set.

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

Table 3 Accuracy results in test phase

Data set	EFS-RPS	FS-SSGA	PS-SSGA	FPS-SSGA	PS-FRW	FS-RST	EIS-RFS	1-NN
Australian	85.12 ± 4.53	85.07 ± 3.49	85.65 ± 2.77	85.36 ± 3.31	84.64 ± 3.15	81.45 ± 4.52	**85.66** ± 2.27	81.45 ± 4.29
Automobile	**82.05** ± 8.55	79.61 ± 6.87	63.78 ± 14.84	69.38 ± 6.33	76.07 ± 7.48	78.97 ± 10.32	61.17 ± 11.58	77.93 ± 6.68
Balance	85.04 ± 6.81	70.89 ± 9.80	86.40 ± 3.08	84.31 ± 4.85	**90.56** ± 1.57	79.04 ± 6.81	85.92 ± 2.62	79.04 ± 6.46
Bands	**74.42** ± 5.45	72.35 ± 6.70	69.77 ± 9.22	64.95 ± 8.69	72.01 ± 9.66	66.61 ± 6.32	64.57 ± 5.91	74.04 ± 6.94
Breast	69.94 ± 7.01	70.98 ± 5.72	70.94 ± 4.63	**73.42** ± 8.34	66.03 ± 6.96	60.97 ± 10.44	69.29 ± 5.73	65.35 ± 6.39
Bupa	63.98 ± 6.82	59.91 ± 10.19	61.14 ± 9.37	62.72 ± 8.40	59.59 ± 9.97	62.51 ± 7.78	**65.72** ± 8.79	61.08 ± 6.88
Car	90.68 ± 1.31	90.68 ± 1.51	89.29 ± 2.55	**93.34** ± 1.37	85.65 ± 3.03	70.02 ± 0.17	91.67 ± 3.31	85.65 ± 1.91
Cleveland	53.81 ± 7.66	51.47 ± 9.47	52.82 ± 4.47	56.13 ± 6.07	**56.82** ± 8.87	52.51 ± 9.49	55.16 ± 5.82	53.14 ± 7.45
Contraceptive	43.31 ± 4.21	41.96 ± 3.57	44.54 ± 4.61	45.15 ± 2.32	44.47 ± 2.57	42.63 ± 3.73	**45.42** ± 5.14	42.77 ± 3.69
Crx	**85.07** ± 3.75	81.16 ± 7.61	84.64 ± 4.22	84.64 ± 5.08	83.04 ± 4.64	81.30 ± 6.28	84.93 ± 5.72	79.57 ± 5.12
Dermatology	95.65 ± 4.07	**96.71** ± 2.85	94.84 ± 4.66	95.36 ± 3.83	95.92 ± 3.43	91.59 ± 3.69	94.81 ± 4.18	95.35 ± 3.64
Ecoli	79.79 ± 7.51	78.90 ± 7.30	80.38 ± 5.69	77.70 ± 5.52	**83.33** ± 6.13	76.58 ± 14.73	82.14 ± 8.42	80.70 ± 7.51
Flare-solar	63.61 ± 3.12	62.76 ± 3.65	64.82 ± 3.37	**67.35** ± 4.12	66.70 ± 3.61	63.23 ± 5.56	66.32 ± 2.94	55.54 ± 3.20
German	**72.00** ± 3.40	69.50 ± 2.68	70.40 ± 3.24	70.10 ± 3.48	70.30 ± 4.57	67.90 ± 3.41	70.80 ± 4.24	70.50 ± 4.25
Glass	71.52 ± 14.45	71.80 ± 14.30	67.10 ± 14.74	71.23 ± 10.64	73.20 ± 14.31	**74.50** ± 13.17	67.35 ± 11.83	73.61 ± 11.91
Haberman	72.81 ± 5.62	72.81 ± 6.15	71.23 ± 5.40	**72.83** ± 5.99	67.65 ± 4.73	65.68 ± 6.58	71.56 ± 7.34	66.97 ± 5.46
Hayes-Roth	**83.93** ± 9.03	83.93 ± 8.33	69.15 ± 11.69	79.80 ± 11.65	75.46 ± 10.29	76.07 ± 14.07	80.86 ± 11.70	35.70 ± 9.11
Heart	78.89 ± 6.77	76.67 ± 6.06	81.11 ± 7.90	**82.59** ± 6.31	82.22 ± 5.18	78.89 ± 6.77	80.74 ± 6.34	77.04 ± 8.89
Hepatitis	81.92 ± 10.03	76.21 ± 7.89	79.33 ± 8.71	80.67 ± 6.13	82.04 ± 10.26	79.50 ± 7.95	**82.58** ± 7.99	82.04 ± 11.09
Housevotes	**96.31** ± 3.65	94.01 ± 4.53	93.79 ± 3.43	94.46 ± 4.37	92.38 ± 5.79	90.78 ± 6.47	94.48 ± 3.67	91.24 ± 5.41
Iris	**96.00** ± 4.66	95.33 ± 4.50	94.67 ± 2.81	94.67 ± 4.22	95.33 ± 5.49	93.33 ± 5.44	**96.00** ± 4.92	93.33 ± 5.16
Led7Digit	63.00 ± 7.54	63.00 ± 6.94	**73.40** ± 2.84	71.40 ± 4.81	63.20 ± 3.43	63.60 ± 5.87	73.20 ± 4.99	40.20 ± 9.48
Lymphography	75.21 ± 9.75	**78.49** ± 9.12	77.92 ± 9.39	74.92 ± 10.79	73.87 ± 9.17	77.38 ± 11.21	77.15 ± 12.15	73.87 ± 8.77
Mammographic	79.42 ± 4.26	75.86 ± 6.07	79.50 ± 3.85	80.15 ± 6.23	79.09 ± 3.80	75.76 ± 4.97	**80.65** ± 4.51	76.38 ± 5.67
Monks	**100.00** ± 0.00	**100.00** ± 0.00	83.53 ± 6.21	98.64 ± 3.07	77.70 ± 5.37	77.91 ± 5.71	**100.00** ± 0.00	77.91 ± 5.42
New thyroid	96.75 ± 2.24	96.30 ± 1.95	**98.16** ± 3.20	96.32 ± 3.60	96.73 ± 3.18	97.23 ± 2.39	96.77 ± 4.83	97.23 ± 2.26
Pima	73.35 ± 5.21	67.70 ± 4.59	72.26 ± 4.44	73.83 ± 3.15	74.49 ± 3.49	70.33 ± 3.71	**74.80** ± 3.71	70.33 ± 3.53
Saheart	69.05 ± 6.69	61.24 ± 3.91	69.27 ± 3.70	67.99 ± 5.69	**71.66** ± 6.12	64.49 ± 4.21	68.82 ± 7.16	64.49 ± 3.99
Sonar	**89.43** ± 6.65	84.62 ± 8.65	75.45 ± 11.74	75.50 ± 12.59	85.57 ± 7.14	81.69 ± 9.83	80.76 ± 7.88	85.55 ± 7.51
Spectfheart	74.56 ± 8.79	74.17 ± 6.34	75.31 ± 5.96	75.34 ± 7.31	**78.36** ± 7.22	70.04 ± 8.00	76.82 ± 7.07	69.70 ± 6.55
Tae	**62.38** ± 13.09	62.37 ± 14.17	54.42 ± 11.63	55.62 ± 13.70	61.08 ± 15.09	60.42 ± 14.29	52.08 ± 11.22	40.50 ± 8.89
Tic-tac-toe	83.20 ± 3.25	**83.51** ± 3.10	78.71 ± 3.36	77.87 ± 5.25	73.07 ± 2.28	73.07 ± 2.70	78.29 ± 5.07	73.07 ± 2.56
Vehicle	**72.70** ± 5.40	70.58 ± 4.92	66.91 ± 4.38	70.92 ± 3.84	68.20 ± 5.65	65.56 ± 6.14	65.37 ± 6.71	70.10 ± 5.90
Vowel	99.19 ± 0.90	99.19 ± 0.80	91.62 ± 3.01	89.60 ± 3.96	99.09 ± 1.00	91.58 ± 4.29	98.81 ± 2.10	**99.39** ± 0.85
Wine	95.52 ± 3.53	94.90 ± 3.30	92.68 ± 7.91	94.93 ± 3.17	95.52 ± 6.84	95.49 ± 4.40	**97.19** ± 5.09	95.52 ± 4.85
Wisconsin	95.85 ± 1.25	95.14 ± 2.62	96.13 ± 2.95	95.86 ± 2.47	**96.86** ± 2.41	95.57 ± 2.73	96.42 ± 1.55	95.57 ± 2.59
Yeast	**54.23** ± 4.01	52.30 ± 3.94	54.18 ± 4.38	53.50 ± 3.77	52.90 ± 3.62	52.23 ± 4.39	53.37 ± 3.36	50.47 ± 3.91
Zoo	**98.33** ± 3.60	95.42 ± 6.00	94.22 ± 7.94	90.72 ± 7.09	98.33 ± 3.60	96.50 ± 4.61	96.39 ± 4.80	92.81 ± 6.57
Average	**79.16** ± 5.65	77.30 ± 5.78	76.56 ± 6.01	77.61 ± 5.83	77.61 ± 5.82	74.81 ± 6.66	78.00 ± 5.86	73.56 ± 5.81
Best result (of 38)	13	4	2	5	6	1	9	1

Table 4 Results of Friedman, Holm and Finner tests

Algorithm	Friedman ranking	Holm *p*-value	Finner *p*-value
FS-RST	7.4211	**0.000002**	**0.000002**
1-NN	7.0263	**0.000030**	**0.000017**
FS-SSGA	6.2368	**0.003437**	**0.001472**
PS-SSGA	5.4605	**0.089441**	**0.031968**
FPS-SSGA	5.1053	0.242954	**0.089710**
PS-FRW	5.0132	0.242954	**0.094809**
EIS-RFS	4.2368	0.544390	0.544390
EFS-RPS	**3.8158**	–	–
Friedman *p*-value		$<10^{-6}$	

Table 5 Average reduction results over features and instances

Data set	Features					Instances				
	EFS-RPS	FS-SSGA	FPS-SSGA	FS-RST	EIS-RFS	EFS-RPS	PS-SSGA	FPS-SSGA	PS-FRW	EIS-RFS
Australian	0.7500	0.8071	0.7929	0.0000	0.1571	0.4288	0.8799	0.8808	0.4032	0.8872
Automobile	0.6760	0.7560	0.7160	0.3265	0.3560	0.0727	0.8444	0.8309	0.0640	0.8531
Bal	0.0000	0.3000	0.0000	0.0000	0.0000	0.5329	0.8686	0.8085	0.5846	0.8464
Bands	0.4842	0.6526	0.5526	0.3750	0.4263	0.3153	0.8472	0.8363	0.3267	0.8689
Bre	0.5444	0.6667	0.7111	0.2111	0.2111	0.0454	0.9441	0.9779	0.0408	0.9476
Bupa	0.4000	0.3667	0.4333	0.1274	0.0000	0.3838	0.8644	0.8644	0.3316	0.8502
Car	0.1667	0.1667	0.1667	0.1667	0.1667	0.0000	0.7681	0.7592	0.0000	0.8279
Cleveland	0.5462	0.7385	0.6077	0.3908	0.0462	0.7141	0.9171	0.9289	0.7352	0.9014
Contraceptive	0.4444	0.4556	0.5889	0.0360	0.0667	0.4133	0.7530	0.7530	0.4463	0.7637
Crx	0.4733	0.5667	0.5533	0.2000	0.1800	0.1685	0.8816	0.8805	0.1750	0.8914
Dermatology	0.5500	0.6676	0.4735	0.4354	0.3854	0.1520	0.9448	0.9414	0.1332	0.9502
Ecoli	0.1571	0.1714	0.1857	0.2286	0.1286	0.3179	0.9077	0.9130	0.3201	0.8882
Flare-solar	0.4778	0.5111	0.5778	0.1556	0.0556	0.8093	0.8391	0.8005	0.7963	0.8122
German	0.6800	0.5150	0.7450	0.1450	0.2350	0.0423	0.7914	0.7928	0.0022	0.8014
Glass	0.4222	0.4444	0.4556	0.0168	0.0444	0.1322	0.8791	0.8791	0.1422	0.8718
Haberman	0.6667	0.6667	0.5333	0.0254	0.0000	0.6560	0.9379	0.9379	0.6385	0.9306
Hayes-Roth	0.2500	0.2500	0.2500	0.1000	0.2500	0.2343	0.8384	0.8452	0.2172	0.8544
Heart	0.6231	0.4538	0.5692	0.1846	0.2308	0.6523	0.9506	0.9230	0.6617	0.9255
Hepatitis	0.6000	0.6684	0.5421	0.4263	0.5368	0.0471	0.9226	0.9355	0.0229	0.9262
Housevotes	0.6313	0.7000	0.7313	0.0188	0.3500	0.0389	0.9410	0.9653	0.0128	0.9387
Iris	0.4000	0.4000	0.4500	0.0000	0.1250	0.3415	0.9481	0.9481	0.1978	0.9511
Led7Digit	0.0143	0.0143	0.0000	0.0143	0.0000	0.5122	0.9071	0.9491	0.5122	0.9416
Lym	0.5111	0.6500	0.6500	0.2611	0.4444	0.0000	0.8994	0.9234	0.0000	0.9257
Mammographic	0.6600	0.5000	0.6200	0.3396	0.0000	0.4644	0.8229	0.7829	0.4722	0.8322
Monks	0.5000	0.5000	0.5333	0.0000	0.5000	0.3922	0.8570	0.9406	0.3760	0.9342
New thyroid	0.3200	0.3000	0.3800	0.0000	0.0600	0.6481	0.9571	0.9571	0.6652	0.9473
Pima	0.5500	0.5750	0.4375	0.0000	0.0000	0.5382	0.8187	0.8187	0.5298	0.7911
Saheart	0.4556	0.6333	0.5778	0.0000	0.0000	0.7245	0.8841	0.8778	0.6857	0.8668
Sonar	0.5767	0.6633	0.6600	0.7183	0.2900	0.0422	0.8595	0.8974	0.0283	0.8899
Spectfheart	0.4977	0.6750	0.6614	0.2750	0.2727	0.5428	0.9426	0.9409	0.5331	0.9497
Tae	0.3000	0.4000	0.2200	0.1183	0.1291	0.1902	0.8727	0.8992	0.1847	0.8764
Tic-tac-toe	0.2444	0.2444	0.2889	0.0000	0.0000	0.0000	0.7917	0.8047	0.0000	0.8655
Vehicle	0.4722	0.4833	0.4778	0.2944	0.2549	0.3256	0.7895	0.7927	0.3082	0.8211
Vowel	0.3077	0.3077	0.3538	0.2894	0.2640	0.0162	0.7201	0.7366	0.0091	0.7552
Wine	0.4846	0.4538	0.4538	0.5231	0.3308	0.6593	0.9538	0.9557	0.5905	0.9451
Wisconsin	0.4556	0.3889	0.3222	0.0000	0.0444	0.3672	0.9027	0.9048	0.4036	0.9103
Yeast	0.1000	0.0875	0.1625	0.1256	0.0375	0.4428	0.7485	0.7485	0.4272	0.7550
Zoo	0.7063	0.7125	0.3750	0.2750	0.2125	0.2349	0.8714	0.8468	0.2150	0.8634
Average	0.4500	0.4872	0.4687	0.1791	0.1787	0.3316	0.8702	0.8731	0.3209	0.8779

Table 4 summarizes the results of the Friedman test, and the post-hoc procedures (Holm and Finner), performed to contrast the results obtained concerning classification accuracy. Average rankings and *p*-value are reported for the Friedman test, and the best (lowest) rank is highlighted in bold. Regarding the post-hoc methods, adjusted *p*-values are provided, highlighting in bold those which represent significant differences (at a 0.1 level of significance).

Table 5 shows the average reduction rates achieved through the application of every data reduction method. On the left-hand side it shows the reduction achieved over the set of features (that is, the ratio of features selected over the original number of features of the problem) and, on the

Table 6 Average time elapsed (training phase), in seconds

Data set	EFS-RPS	FS-SSGA	PS-SSGA	FPS-SSGA	PS-FRW	FS-RST	EIS-RFS
Australian	111.35	161.39	79.16	48.14	21.53	0.70	82.54
Automobile	16.80	50.27	14.40	8.55	0.40	0.36	30.48
Bal	29.27	88.56	38.71	38.33	0.21	0.03	54.44
Bands	85.28	287.05	75.96	63.93	13.09	1.41	116.68
Bre	13.77	43.73	8.86	5.67	0.13	0.09	12.32
Bupa	13.60	32.65	13.70	11.33	1.25	0.04	20.71
Car	475.93	1619.67	442.19	520.30	1.17	0.20	560.60
Cleveland	20.25	33.37	11.89	9.04	1.61	0.10	19.29
Contraceptive	352.47	704.06	348.66	306.52	9.09	1.32	316.30
Crx	114.32	220.59	79.72	70.56	6.73	0.46	86.38
Dermatology	66.46	186.50	35.90	27.80	2.33	0.30	60.20
Ecoli	14.63	37.50	10.92	11.17	1.36	0.05	20.68
Flare-solar	183.55	349.09	160.00	123.76	0.95	0.01	183.44
German	448.56	591.00	252.59	167.51	1.63	2.07	304.94
Glass	6.95	15.93	5.39	5.18	0.53	0.05	10.30
Haberman	7.39	13.63	7.09	6.06	0.22	0.01	9.41
Hayes-Roth	1.79	5.00	2.68	2.52	0.09	0.02	3.86
Heart	15.19	32.98	8.03	7.01	1.18	0.06	14.57
Hepatitis	7.89	13.08	3.83	3.21	0.11	0.04	8.50
Housevotes	50.86	82.91	24.98	17.38	0.31	0.02	39.42
Iris	1.64	5.22	2.44	2.33	0.17	0.02	4.40
Led7Digit	36.43	88.31	25.05	28.87	0.18	0.01	40.50
Lym	6.44	11.77	3.97	3.23	0.09	0.02	8.14
Mammographic	123.48	205.34	116.67	77.57	1.62	0.20	127.75
Monks	18.91	46.18	20.72	14.22	0.16	0.02	27.92
New thyroid	3.93	11.76	3.68	3.63	0.40	0.01	8.03
Pima	88.43	175.95	85.38	72.09	17.65	0.29	96.68
Saheart	34.33	62.64	25.98	22.45	4.25	0.15	33.45
Sonar	39.49	66.79	17.12	13.86	0.78	0.40	136.71
Spectfheart	38.13	80.53	15.14	15.25	3.98	0.27	40.33
Tae	2.48	9.41	2.42	2.37	0.11	0.03	3.27
Tic-tac-toe	144.63	348.57	150.31	132.37	0.59	0.06	176.75
Vehicle	409.15	671.39	220.63	183.52	47.11	6.13	495.09
Vowel	495.70	867.64	270.26	241.58	60.75	4.39	461.71
Wine	3.77	14.80	3.62	3.34	0.48	0.03	7.67
Wisconsin	70.02	159.05	55.66	50.62	2.81	0.10	73.61
Yeast	295.60	825.97	354.55	364.96	109.57	1.47	420.58
Zoo	2.43	5.12	2.52	2.46	0.05	0.01	5.31
Average	101.35	216.46	78.97	70.75	8.28	0.55	108.50

right-hand side, the reduction achieved over the set of instances (the ratio of prototypes selected over the total number of instances of the original training set).

Finally, Table 6 reports the time elapsed for the methods in training phase (in seconds).[4] Note that the running times in the test phase are not reported since they are too low to show interesting differences. Furthermore, the efficiency in the test phase is already reflected by the reduction rates achieved (the higher the reduction rates are, the less running time will be needed).

We can draw the following conclusions:

- The new hybrid approach, EFS-RPS, obtains the best results in accuracy. As Table 3 shows, it achieves the

[4] The experiments have been carried out on a machine with a Dual Core 3,20 GHz processor and 2GB of RAM, running under the Fedora 4 operating System.

best average result, and the best result for 13 out of 38 data sets. The introduction of fuzzy rough set theory to improve the evolutionary techniques turns out to be very effective when its results are compared with the basic techniques considered in isolation.

- All the data reduction methods considered in the study outperform the basic 1-NN accuracy, PS-FRW being the best non-hybrid technique when the average ranking is considered. FS and PS, when performed in an effective way—such as using the preprocessing methods considered in the study—are able to improve the accuracy of the 1-NN classifier (sometimes providing a substantial improvement, such as in the case of EFS-RPS, with an average accuracy increase of more than 5 %. This aspect is further reflected by the starplot represented in Fig. 1).
- The hybridization performed to design EFS-RPS has not damaged the reduction power of the base techniques on which it is based: as Table 5 reports, its reduction rate over features is similar to the one obtained by FS-SSGA, the evolutionary FS method which guides its search; its reduction rate over instances is also similar to the one obtained by PS-FRW, its basic PS inner procedure. Hence, the hybridization has shown to be effective for increasing the accuracy of the preprocessing technique without damaging the reduction capabilities of the standalone procedures.
- Concerning running time, Table 6 shows that EFS-RPS has an average behavior: it is somewhat slower than the evolutionary methods with high prototype reduction power (PS-SSGA, FPS-SSGA), but faster than the evolutionary methods that focus only on performing FS (FS-SSGA). It is also faster than the hybrid preprocessing method considered in the comparison (EIS-RFS).

The statistical study, performed to contrast the results obtained in accuracy, confirms our analysis: the Friedman

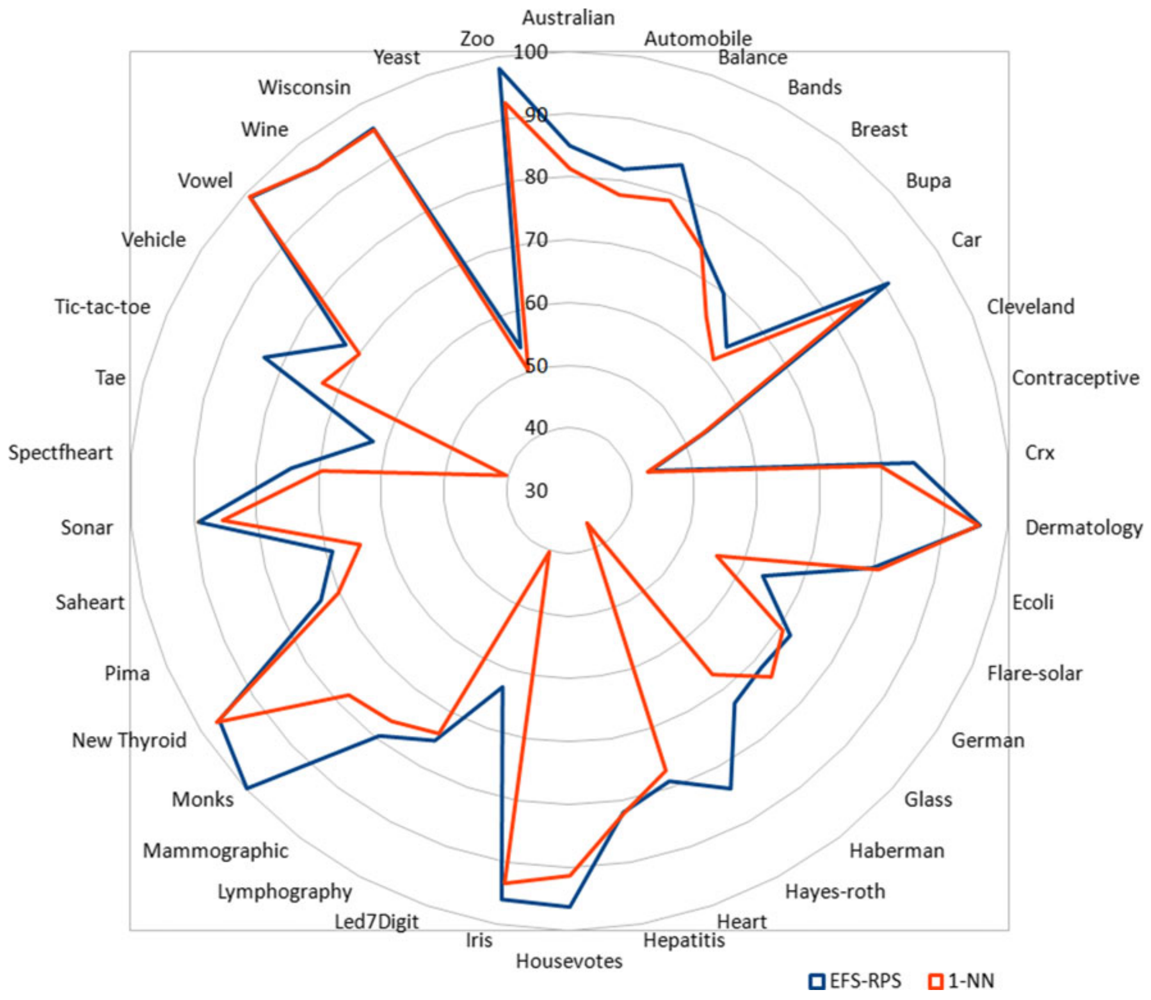


Fig. 1 Starplot depicting the enhancement in accuracy of the 1-NN classifier when EFS-RPS is used for preprocessing the data. The differences between the areas of the star represent absolute differences between the precision of both classifiers classifying unseen instances in test phase

test detects significant differences among the methods (with a p-value $<10^{-6}$) and highlights EFS-RPS as the best performing one, with a rank of 3.8158 (the lowest; see Table 4). The Holm test establishes the existence of significant differences between EFS-RPS and FS-RST, 1-NN, FS-SSGA and PS-SSGA, which are further expanded by the Finner test, marking as significant the comparisons between EFS-RPS and FPS-SSGA and PS-FRW. Only the differences between EFS-RPS and EIS-RFS are not strong enough to be marked as significant.

Differences between the behavior of EIS-RFS and EFS-RPS can be found if a smaller subset of problems is considered (instead of the large set of domains chosen in this study); thus, a suitable choice may depend upon the specific problem to tackle. However, we can point out that the small difference found between both hybrid methods can be caused due to the lower variation in the possible feature sets selected by EIS-RPS, which is not an issue for the evolutionary FS part of EFS-RPS.

In summary, EFS-RPS can be highlighted for being a highly accurate method for performing dual data reduction (including FS and PS) for the 1-NN classifier. It improves the accuracy of evolutionary and fuzzy rough set based data reduction approaches without losing reduction power and without increasing the time complexity of the procedures considered. Therefore, it is a competent method for performing data reduction which can be applied for improving the performance of the 1-NN in any standard supervised classification domain.

5 Conclusions

In this work, we have proposed a new approach based on fuzzy rough sets and evolutionary algorithms for performing a simultaneous process of FS and PS. This data reduction process is specifically designed to improve the performance of the 1-NN classifier, both regarding test accuracy and computational complexity.

The results achieved by EFS-RPS in the experimental study performed have shown that it offers the best results among all the related techniques selected for the comparison; that is, the hybrid approach outperforms those methods based only in either evolutionary techniques or fuzzy rough set ones. Nonparametric statistical procedures have been used to contrast this results, supporting the conclusions arrived at.

These promising results allow us to point out further extensions of the EFS-RPS model, and new directions of research related. One of them would be to test the behavior of the model when considering other base classifiers. That is, to apply the model and preprocess the data using other classifiers different than 1-NN. This extension would put our approach in the field of Training Set Selection (see (Kim 2006; Cano et al. 2007, 2008) for some promising applications, and (Derrac et al. 2010b; García-Pedrajas 2011) for two reviews on evolutionary approaches to the field), analogous to PS, providing more generality in the range of domains in which EFS-RPS can be applied.

Another interesting trend of research can be focused on particular traits of the data. Imbalanced data sets (He and Garcia 2009) pose a problem nowadays in many applications of research. This tough problem requires the definition of specific methods, measures and evaluation procedures; however, the application of evolutionary preprocessing methods for nearest neighbor classifiers and rough set theory with success is still possible (see García and Herrera 2009; Ramentol et al. 2012, respectively). Hence, further research on extensions of EFS-RPS could be focused on obtaining a new version of the model, suitable for tackling imbalanced domains.

Supported by the Spanish Ministry of Science and Technology (Project TIN2011-28488) and by the Regional Project P10-TIC-06858. J. Derrac holds an FPU scholarship from the Spanish Ministry of Education. C. Cornelis holds a Ramón y Cajal research contract from the Spanish Ministry of Science and Innovation.

References

Aha DW, Kibler D, Albert MK (1991) Instance-based learning algorithms. Mach Learn 6:37–66

Alcalá-Fdez J, Sánchez L, García S, del Jesus MJ, Ventura S, Garrell JM, Otero J, Romero C, Bacardit J, Rivas VM, Fernández JC, Herrera F (2008) KEEL: a software tool to assess evolutionary algorithms for data mining problems. Soft Comput 13(3):307–318

Alcalá-Fdez J, Fernández A, Luengo J, Derrac J, García S, Sánchez L, Herrera F (2011) Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. J Mult Valued Log Soft Comput 17(2–3):255–287

Almuallim H, Dietterich T (1991) Learning with many irrelevant features. In: Proceedings of the 9th national conference on artificial intelligence, vol 2, Anaheim, CA, USA, July 14–19, The MIT Press, pp 547–552

Alpaydin E (2010) Introduction to machine learning, 2nd edn. The MIT Press, Cambridge

Bell G, Hey T, Szalay A (2009) Beyond the data deluge. Science 323:1297–1298

Cano JR, Herrera F, Lozano M (2003) Using evolutionary algorithms as instance selection for data reduction in KDD: An experimental study. IEEE Trans Evol Comput 7(6):561–575

Cano JR, Herrera F, Lozano M (2007) Evolutionary stratified training set selection for extracting classification rules with trade-off precision-interpretability. Data Knowl Eng 60:90–100

Cano JR, Herrera F, Lozano M, García S (2008) Making CN2-SD subgroup discovery algorithm scalable to large size data sets using instance selection. Expert Syst Appl 35:1949–1965

Casillas J, Cordon O, Del Jesus MJ, Herrera F (2001) Genetic feature selection in a fuzzy rule-based classification system learning process for high-dimensional problems. Inf Sci 136:135–157

Chen Y, Garcia EK, Gupta MR, Rahimi A, Cazzanti L (2009) Similarity-based classification: concepts and algorithms. J Mach Learn Res 10:747–776

Cornelis C, Jensen R, Hurtado G, Slezak D (2010) Attribute selection with fuzzy decision reducts. Inf Sci 180:209–224

Cover TM, Hart PE (1967) Nearest neighbor pattern classification. IEEE Trans Inf Theory 13(1):21–27

De Cock M, Cornelis C, Kerre EE (2007) Fuzzy rough sets: The forgotten step. IEEE Trans Fuzzy Syst 15(1):121–130

Derrac J, García S, Herrera F (2010a) IFS-CoCo: Instance and feature selection based on cooperative coevolution with nearest neighbor rule. Pattern Recognit 43(6):2082–2105

Derrac J, García S, Herrera F (2010b) A survey on evolutionary instance selection and generation. Int J Appl Metaheur Comput 1(1):60–92

Derrac J, Cornelis C, García S, Herrera F (2012) Enhancing evolutionary instance selection algorithms by means of fuzzy rough set based feature selection. Inf Sci 186(1):73–92

Destercke S (2012) A k-nearest neighbours method based on imprecise probabilities. Soft Comput 16(5):833–844

Dubois D, Prade H (1990) Rough fuzzy sets and fuzzy rough sets. Int J General Syst 17:191–209

Eiben AE, Smith JE (2003) Introduction to Evolutionary Computing. Natural Computing, Springer-Verlag, Berlin

Eshelman LJ (1991) The CHC adaptive search algorithm: how to have safe search when engaging in nontraditional genetic recombination. In: Rawlins GJE (ed) Foundations of genetic algorithms, Morgan Kaufmann, San Mateo, pp 265–283

Ferrandiz S, Boullé M (2010) Bayesian instance selection for the nearest neighbor rule. Mach Learn 81(81):229–256

Franco A, Maltoni D, Nanni L (2010) Data pre-processing through reward-punishment editing. Pattern Anal Appl 13:367–381

Frank A, Asuncion A (2010) UCI machine learning repository. http://archive.ics.uci.edu/ml

Freitas AA (2002) Data mining and knowledge discovery with evolutionary algorithms. Springer-Verlag, Berlin

García S, Herrera F (2008) An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons. J Mach Learn Res 9:2677–2694

García S, Herrera F (2009) Evolutionary undersampling for classification with imbalanced datasets: Proposals and taxonomy. Evol Comput 17(3):275–306

García S, Cano JR, Herrera F (2008) A memetic algorithm for evolutionary prototype selection: A scaling up approach. Pattern Recognit 41(8):2693–2709

García S, Fernández A, Luengo J, Herrera F (2009) A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability. Soft Comput 13(10):959–977

García S, Fernández A, Luengo J, Herrera F (2010) Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power. Inf Sci 180:2044–2064

García S, Derrac J, Cano JR, Herrera F (2012a) Prototype selection for nearest neighbor classification: taxonomy and empirical study. IEEE Trans Pattern Anal Mach Intell 34(3):417–435

García S, Luengo J, Sáez JA, López V, Herrera F (2012b) A survey of discretization techniques: taxonomy and empirical analysis in supervised learning. IEEE Trans Knowl Data Eng (in press)

García-Pedrajas N (2011) Evolutionary computation for training set selection. Wiley Interdiscip Rev Data Min Knowl Dis 1(6):512–523

García-Pedrajas N, Romero JA, Ortiz-Boyer D (2010) A cooperative coevolutionary algorithm for instance selection for instance-based learning. Mach Learn 78:381–420

Ghosh A, Jain LC (eds) (2005) Evolutionary computation in data mining. Springer-Verlag, Berlin

Gil-Pita R, Yao X (2008) Evolving edited k-nearest neighbor classifiers. Int J Neural Syst 18(6):1–9

Gonzalez A, Perez R (2001) Selection of relevant features in a fuzzy genetic learning algorithm. IEEE Trans Syst Man Cybern 31(3): 417–425

Guyon I, Elisseeff A (2003) An introduction to variable and feature selection. J Mach Learn Res 3:1157–1182

Guyon I, Gunn S, Nikravesh M, Zadeh LA (eds) (2006) Feature extraction: foundations and applications. Springer, Berlin

Hart PE (1968) The condensed nearest neighbour rule. IEEE Trans Inf Theory 18(5):515–516

He H, Garcia EA (2009) Learning from imbalanced data. IEEE Trans Knowl Data Eng 21:1263–1284

He Q, Wu C (2011) Membership evaluation and feature selection for fuzzy support vector machine based on fuzzy rough sets. Soft Comput 15(6):1105–1114

Ho SY, Liu CC, Liu S (2002) Design of an optimal nearest neighbor classifier using an intelligent genetic algorithm. Pattern Recognit Lett 23(13):1495–1503

Inza I, Larrañaga P, Sierra B (2001) Feature subset selection by bayesian networks: a comparison with genetic and sequential algorithms. Int J Approx Reason 27:143–164

Ishibuchi H, Nakashima T (1998) Evolution of reference sets in nearest neighbor classification. In: Second Asia-Pacific conference on simulated evolution and learning on simulated evolution and learning (SEAL'98). Lecture notes in computer science, vol 1585, pp 82–89

Ishibuchi H, Nakashima T, Nii M (2001) Genetic-algorithm-based instance and feature selection. In: Liu H, Motoda H (eds) Instance selection and construction for data mining, Kluwer Academic Publishers, Dordrecht, pp 95–112

Jensen R, Cornelis C (2010) Fuzzy-rough instance selection. In: Proceedings of the WCCI 2010 IEEE world congress on computational intelligence, IEEE congress on fuzzy logic, Barcelona, Spain, pp 1776–1782

Jensen R, Shen Q (2007) Fuzzy-rough sets assisted attribute selection. IEEE Trans Fuzzy Syst 15(1):73–89

Jensen R, Shen Q (2009) New approaches to fuzzy-rough feature selection. IEEE Trans Fuzzy Syst 17(4):824–838

Kim K (2006) Artificial neural networks with evolutionary instance selection for financial forecasting. Expert Syst Appl 30:519–526

Kira K, Rendell L (1992) A practical approach to feature selection. In: Proceedings of the 9th international workshop on machine learning, Aberdeen, Scotland UK, pp 249–256

Kohavi R, John G (1997) Wrappers for feature selection. Artif Intell 97:273–324

Kuncheva LI (1995) Editing for the *k*-nearest neighbors rule by a genetic algorithm. Pattern Recognit Lett 16:809–814

Kuncheva LI, Jain L (1999) Nearest neighbor classifier: simultaneous editing and descriptor selection. Pattern Recognit Lett 20:1149–1156

Kusunoki Y, Inuiguchi M (2010) A unified approach to reducts in dominance-based rough set approach. Soft Comput 14(5):507–515

Liu H, Motoda H (eds) (1998) Feature selection for knowledge discovery and data mining. The Springer international series in engineering and computer science, Springer, Berlin

Liu H, Motoda H (eds) (2001) Instance selection and construction for data mining. The Springer international series in engineering and computer science, Springer, Berlin

Liu H, Motoda H (eds) (2007) Computational methods of feature selection. Chapman & Hall/Crc data mining and knowledge discovery series, Chapman & Hall/Crc, London

Liu H, Yu L (2005) Toward integrating feature selection algorithms for classification and clustering. IEEE Trans Knowl Data Eng 17(3):1–12

Mjolsness E, DeCoste D (2001) Machine learning for science: state of the art and future prospects. Science 293:2051–2055

Oh IS, Lee JS, Moon BR (2004) Hybrid genetic algorithms for feature selection. IEEE Trans Pattern Anal Mach Intell 26:1424–1437

Pappa GL, Freitas AA (2009) Automating the design of data mining algorithms: an evolutionary computation approach. Natural computing. Springer, Berlin

Pawlak Z (1982) Rough sets. Int J Comput Inf Sci 11(5):341–356

Pawlak Z (1991) Rough sets: theoretical aspects of reasoning about data. Kluwer Academic Publishing, Dordrecht

Pawlak Z, Skowron A (2007a) Rough sets: some extensions. Inf Sci 177(1):28–40

Pawlak Z, Skowron A (2007b) Rudiments of rough sets. Inf Sci 177:3–27

Pyle D (1999) Data preparation for data mining. The Morgan Kaufmann series in data management systems. Morgan Kaufmann, Menlo Park

Quirino T, Kubat M, Bryan NJ (2010) Instinct-based mating in genetic algorithms applied to the tuning of 1-nn classifiers. IEEE Trans Knowl Data Eng 22(12):1724–1737

Radzikowska A, Kerre E (2002) A comparative study of fuzzy rough sets. Fuzzy Sets Syst 126:137–156

Ramentol E, Verbiest N, Bello R, Caballero Y, Cornelis C, Herrera F (2012) SMOTE-FRST: a new resampling method using fuzzy rough set theory. In: 10th International FLINS conference on uncertainty modelling in knowledge engineering and decision making (to appear)

Rokach L (2008) Genetic algorithm-based feature set partitioning for classification problems. Pattern Recognit 41:1676–1700

Saeys Y, Inza I, Larrañaga P (2007) A review of feature selection techniques in bioinformatics. Bioinformatics 19:2507–2517

Shakhnarovich G, Darrell T, Indyk P (eds) (2006) Nearest-neighbor methods in learning and vision: theory and practice. The MIT Press, Cambridge

Sheskin DJ (2011) Handbook of parametric and nonparametric statistical procedures, 5th edn. Chapman & Hall/CRC, London

Shie J, Chen S (2008) Feature subset selection based on fuzzy entropy measures for handling classification problems. Appl Intell 28: 69–82

Stracuzzi D, Utgoff P (2004) Randomized variable elimination. J Mach Learn Res 5:1331–1362

Triguero I, García S, Herrera F (2010) IPADE: Iterative prototype adjustment for nearest neighbor classification. IEEE Trans Neural Netw 21(12):1984–1990

Triguero I, Derrac J, García S, Herrera F (2012) A taxonomy and experimental study on prototype generation for nearest neighbor classification. IEEE Trans Syst Man Cybern Part C Appl Rev 42(1):86–100

Tsang E, Chen D, Yeung D, Wang X, Lee JT (2008) Attributes reduction using fuzzy rough sets. IEEE Trans Fuzzy Syst 16(5): 1130–1141

Weinberger K, Saul L (2009) Distance metric learning for large margin nearest neighbor classification. J Mach Learn Res 10: 207–244

Whitley LD (1989) The genitor algorithm and selection pressure: Why rank-based allocation of reproductive trials is best. In: Proceedings of the 3rd international conference on genetic algorithms, vol 2, Fairfax, Virginia, USA, June 1989, Morgan Kaufmann, pp 116–123

Wilson DL (1972) Asymptotic properties of nearest neighbor rules using edited data. IEEE Trans Syst Man Cybern 2(3):408–421

Wilson DR, Martinez TR (2000) Reduction techniques for instance-based learning algorithms. Mach Learn 38(3):257–286

Witten IH, Frank E, Hall MA (2011) Data mining: practical machine learning tools and techniques, 3rd edn. Morgan Kaufmann series in data management systems. Morgan Kaufmann, Menlo Park

Wu X, Kumar V (eds) (2009) The top ten algorithms in data mining. Data mining and knowledge discovery. Chapman & Hall/CRC, London

Zadeh LA (1965) Fuzzy sets. Inf Control 8(3):338–353

Zhai J (2011) Fuzzy decision tree based on fuzzy-rough technique. Soft Comput 15(6):1087–1096

4. Fuzzy Nearest Neighbor Classification

The journal papers associated to this part are:

4.1 Fuzzy Nearest Neighbor Algorithms: Taxonomy, Experimental analysis and Prospects

- J. Derrac, S. García, F. Herrera, Fuzzy Nearest Neighbor Algorithms: Taxonomy, Experimental analysis and Prospects. **Submitted to Information Sciences**.
  - Status: **Submitted**.
  - Impact Factor (JCR 2011): 2.833.
  - Subject Category: Computer Science, Information Systems. Ranking 9 / 135 (**Q1**).

Manuscript Number: INS-D-12-1683

Title: Fuzzy Nearest Neighbor Algorithms: Taxonomy, Experimental analysis and Prospects

Article Type: Full length article

Keywords: Fuzzy Nearest Neighbor; Nearest Neighbor; Taxonomy; Experimental Framework; Classification

Corresponding Author: Mr. Joaquín Derrac, M. D.

Corresponding Author's Institution: University of Granada

First Author: Joaquín Derrac, M. D.

Order of Authors: Joaquín Derrac, M. D.; Salvador García, Ph. D.; Francisco Herrera, Ph. D.

Suggested Reviewers: M.A. Chikh
mea_chihk@mail.univ-tlemcen.dz
Expert in fuzzy K-NN

M. Hayat
maqsood.hayat@pieas.edu.pk
Expert in fuzzy K-NN

J.Y. Jiang
leesj@mail.ee.nsysu.edu.tw
Expert in fuzzy K-NN

Department of Computer Science and Artificial Intelligence, CITIC-UGR (Research Center on Information and Communications Technology)

University of Granada

Granada, Spain 18071

Tuesday, 4 December 2012

Professor Witold Pedrycz

Information Sciences

Dear Professor Witold:

Please find enclosed a manuscript entitled: "Fuzzy Nearest Neighbor Algorithms: Taxonomy, Experimental analysis and Prospects" which I am submitting for exclusive consideration of publication as an article in Information Sciences. Therefore, this is the authors' original work and has not been published nor has it been submitted simultaneously elsewhere. Moreover, all authors have checked the manuscript and have agreed to the submission.

The paper performs a review on the field of fuzzy nearest neighbor classification. Thus is it focused on the use of fuzzy sets theory and several extensions to improve the performance of the nearest neighbor classifier. In addition to the survey, a taxonomy is proposed and a full experimental study is carried out comparing the performance of the methods with several state of the art crisp nearest neighbor classifiers.

Thank you for your consideration of my work! Please address all correspondence concerning this manuscript to me at University of Granada and feel free to correspond with me by e-mail (jderrac@decsai.ugr.es).

Yours sincerely,

Joaquín

Fuzzy Nearest Neighbor Algorithms: Taxonomy, Experimental analysis and Prospects

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

[a] *Dept. of Computer Science and Artificial Intelligence, CITIC-UGR (Research Center on Information and Communications Technology). University of Granada, 18071 Granada, Spain*
[b] *Dept. of Computer Science. University of Jaén, 23071 Jaén, Spain*

Abstract

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

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

Keywords: Fuzzy Nearest Neighbor, Nearest Neighbor, Taxonomy, Experimental Framework, Classification.

---

*Corresponding author. Tel.: +34 958 240598; fax: +34 958 243317.
Email addresses: `jderrac@decsai.ugr.es` (Joaquín Derrac), `sglopez@ujaen.es` (Salvador García), `herrera@decsai.ugr.es` (Francisco Herrera)

1. Introduction

The nearest neighbor (NN) rule is a nonparametric method for pattern classification [43] based on instances [1]. Introduced by Fix and Hodges in 1951 [28], the NN rule gained considerable popularity after 1967, when some of its formal properties were described by Cover and Hart [23]. Cover's work was the key milestone of a subject which now has become a lively research field for many researchers in pattern recognition and machine learning [4, 95] areas: the study and development of one of the top ten algorithms in data mining [96].

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

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

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

The aforementioned drawbacks have been analyzed extensively by the research community. As a result, many approaches have been proposed regarding, for example, the computation of similarity measures [16], the optimum choice of the k parameter [59], the definition of weighting schemes for patterns and attributes [91, 50], the adaptation of the algorithm to data [42], the development of fast and approximate versions of the NN rule, devised to quicken the computation of the nearest neighbors [36, 5], and the reduction of the training data [30, 86, 22, 26].

Fuzzy Sets Theory (FST) [102] has been the basis of a remarkable number of these approaches. Several traits of the k-NN classifier are inherently crisp, and therefore prone to be extended and improved by incorporating the use of fuzzy sets. These traits include the definition of a degree of membership of the instances to a class, the use of similarity measures as a means of fuzzyfying the contribution of each neighbor in the decision rule, the set up of the k parameter and even the definition of new ways of combining the votes of the nearest neighbors.

The study of these characteristics has been tackled considering FST, various extensions and related approaches including fuzzy rough sets [24], intuitionistic fuzzy sets [7], possibilistic theory [103] and type-2 fuzzy sets [57].

Supported by the former approaches, the development of the field has raised since 1983 and 1985, with the first works in the area published by Jóźwik [55] and Keller et al [58]. New approaches regarding both improvements of the k-NN model and applications to real-world problems have been proposed until nowadays, arousing the attention of many researchers and practitioners.

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

Moreover, a full experimental framework, including a set of well-known publicly accessible and representative supervised classification problems, is offered. This framework offers implementations of the essential methods for reference, and suggest a statistical methodology based on nonparametric procedures, which should be enough for providing a rigorous confirmation of the differences reported in most of cases. The framework's description is concluded with a case of use comparing fuzzy k-NN based methods with a set of representative crisp k-NN based approaches. After the analysis of results, the paper is finished with the suggestion of several interesting research trends that remain opened within the topic.

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

2. Fuzzy nearest neighbor algorithms

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

This section is devoted to survey relevant works in these directions, describing the key elements of each approach. After the survey, several common properties of the methods are identified and described. These properties are used to characterize the main approaches surveyed, providing with an insight on the existing differences in the design of the methods.

2.1. A survey on fuzzy nearest neighbor classification

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

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

2.1.1. Nearest neighbor algorithms based on fuzzy sets theory

The first proposal of a fuzzy nearest neighbor classifier was presented by Jóẃik [55] in 1983. This classifier, JFKNN, is an improved version of the standard k-NN. It is based on a learning scheme of class memberships, providing to each training instance a membership array which defines its fuzzy membership to each class. After the learning process, the final classification is performed similarly to k-NN, but every neighbor uses its membership array for the voting rule, instead of just giving one vote as in the crisp k-NN.

Two years later, Keller et al. [58] proposed what now has become the major reference in this field (currently with more than 450 citations in the ISI Web of Science). FuzzyKNN, the classifier described in this work, has been the baseline of many advanced methods hybridizing FST and k-NN classifiers. Furthermore, there is plenty of applications in many fields of research based in this model, mainly due to its good behavior when tackling supervised learning problems.

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

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

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

[1]One of them is the 'crisp' one: to assign a membership of 1 to the class of the instance in the original data, and 0 to the rest of classes

[2]*kInit* is usually set to a value between $(3, 10)$

where v_c are the number of neighbors found belonging to class c, and ω is the class of $\mathbf{x}_i$ in the original data.

The effect of Equation 1 is that instances close to the center of the classes kept their original crisp memberships[3] but instances close to the boundaries between classes will spread half of their membership between the neighbors' classes. However, it is interesting to note that the 0.51 and 0.49 coefficients still ensure that the largest membership will be assigned to the ω class, regardless of the neighboring instances [4]. Also, the sum of the memberships to all classes will be always 1.

- A modified voting rule in which each neighboring instance votes for every class, using the memberships learned during the training phase. These votes are weighted according to the inverse of the distance to the instance to be classified, and finally all votes are added. The final class, $\hat{\omega}$, is obtained as the class with the greatest combined vote.

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

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

All this classical approaches are reviewed by Yang et al. [98], whose work also includes a theoretical proof that the FuzzyKNN rule is bounded above by twice the Bayes risk, extending the results of Cover and Hart for the k-NN rule [23]. The convergence properties of this error are also studied in [99], extending the original review.

[3] 1 to their original class ω and 0 to the rest

[4] 0.51 or more to their original class ω and 0.49 or less to the rest

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

The most common approach is the modification of the way in which membership weights are computed. In [39], the VWFKNN classifier sets weights according to to the standard deviation of the neighbors' class membership values. In this way, weights can model a discriminant function identifying the different classes of the classification problem. Another example is [78], where Pham et al. designed a method based on a kriging system for obtaining the membership weights.

Another trend of work is focused in the modification of the computation of distances. In [61] the distances between the instances are modified depending on its typicalness. Using expert knowledge (expert council interviewing for a medical problem, in this case) fuzzy decision rules are derived in order to obtain an accurate nearest neighbor classifier. Fuzzy distances, represented by fuzzy numbers, are also considered in [72]. In this second case, the introduction of fuzzy distances allows Mitchell et al. method to adapt automatically the value of k according to the local density of the training instances.

Other extensions are focused in the enhancement of the FuzzyKNN by the optimization of the $kInit$ and m parameters. For example, GAFuzzyKNN [45] employs a genetic algorithm for optimizing both values. A parallel implementation of a genetic algorithm designed for this task is also presented in [82].

2.1.2. Interval type-2 fuzzy sets based approach

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

2.1.3. Possibilistic k-NN methods

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

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

Possibilistic instance based learning is also analyzed in [48]. The paper is focused in the development of a theoretical possibilistic framework, linking its properties with those of nearest neighbor classification and analyzing advanced concepts about uncertainty in nearest neighbor classification, similarity measures, noise and outliers detection, and incomplete information management. It also present a classifier based in this concepts, PosIBL, which does not need the specification of the k parameter of the classic k-NN rule.

2.1.4. Intuitionistic k-NN methods

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

In [37] the IFSKNN classifier was proposed. In this algorithm, a value of membership is computed for each instance, as the distance to the mean of the class. Then, the nonmembership value is computed in a similar way, as the distance to the nearest mean of the rest of the classes. This enables to represent typical instances, near to the center of the classes, with a high degree of membership whereas noisy instances, nearer to the center of different classes, will be assigned with a high degree of non membership [5]. The classification is completed using the membership and nonmembership to modify the distances computed by a k-NN classifier.

A second approach using intuitionistic fuzzy sets was presented in [62]. IF-KNN considers the nonmembership degree as the opposite of the membership of each instance to the class, and employs both values to determine the contribution of each neighbor's vote to the final classification.

Finally, in [38] the IFV-NP classifier was proposed. It is a prototypical version of IFSKNN, in which after obtaining the prototypes a procedure is carried out to adjust the degrees of membership and nonmembership in accordance with the distances to the center of the classes.

2.1.5. Preprocessing approaches via data reduction

Preprocessing methods have become an effective way of enhancing the performance of general nearest neighbor classifiers. Among them, prototype selection [30] and prototype generation [86] fields have inspired the first preprocessing methods for fuzzy nearest neighbor classifiers.

Regarding prototype selection, the FENN classifier [100] is based on Wilson's editing rule for k-NN [93]: All instances in the training set are checked and those whose classification by the FuzzyKNN rule does not agree with its original class are removed. CFKNN [104] is also inspired in a classic method, the Hart's condensing rule [40], but using the sample fuzzy entropy to determine whether an instance is finally removed or kept. Another representative example is the PFKNN method [6], which firstly build a set of prototypes representing the border points of different clusters in the data and then adds to this reference set those instances which could be misclassified. The algorithm concludes with a pruning phase in which non-relevant prototypes are discarded.

Finally, it is also possible to find prototype generation methods such as, for example, the Gayar et al. method [34], which describes the use of Fuzzy C-Means for obtaining the membership weights of prototypes generated in an iterative way.

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

2.1.6. Fuzzy rough sets based approaches

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

A first proposal, FRNNA, was presented in [12]. This classifier incorporates the lower and upper approximations of the memberships to the decision rule, in an effort to deal both with fuzzy uncertainties and rough uncertainties. A second proposal - FRNN [83] - develops further this aspect, associating fuzzy uncertainties to the existing overlapping between classes and rough uncertainties to the lack of a proper number of features to describe the data. Another main feature of this method is that it does not require to fix a k value for the classification rule.

Fuzzy-rough nearest neighbor classification is developed in [52]. In these works, the FRNN-FRS and FRNN-VQRS classifiers are described. They employ fuzzy rough sets and vaguely quantified rough sets, respectively. The first classifier is presented as an improvement of FRNN, whereas in the second one vaguely quantified rough sets are introduced to reduce the sensitivity of the classifier to noise. Finally, a further step in fuzzy-rough nearest neighbor classification is presented in [79], where Qu et al. presents an approach to hybridize kernel-based classification with fuzzy rough sets.

2.1.7. Further extensions

Besides the flourishing number of proposals appeared in the literature, presenting a rich variety of fuzzy nearest neighbor classifiers, the joint use of fuzzy sets and the nearest neighbor classifier has gone beyond. Several works, focused either in the application of fuzzy nearest neighbor rules to tackle different problems (instead of classification) or in other ways of combining FST and k-NN. This subsection surveys some of the most interesting approaches:

- The success of FuzzyKNN and other fuzzy nearest neighbor classifiers has inspired similar techniques for incremental data problems [85] (when the full training set is not available at the training phase), outliers detection in temporal series [75], regression [81], semi-supervised learning for monitoring evolving systems [41], multi-label text categorization [53] or low quality data problems [69].

- FST has become an interesting tool for the enhancement of the classic k-NN classifier throughout data preprocessing approaches. Some remarkable examples include [49, 73] where an evolutionary instance selection method is presented and extended. The method is enhanced through the transformation of the instances into circular-conic fuzzy rules, which are finally used to train the classifiers. Instance selection is also the focus of [51] and [56], where two different instance selection methods based on fuzzy rough sets are described.

  Besides preprocessing approaches, several works have also investigated further ways of extending the nearest neighbor classifiers . For example, in [97], a theoretical description of a lineal programming method is provided. This method is aimed to the design of ordered weighted average operators for the decision rule of k-NN. A different approach is presented in [65], which includes an approximated nearest neighbor classifier [5] as a fast classification model, based on fuzzy rough sets.

- Finally, FuzzyKNN has also been considered as a part of larger and more complex classification algorithms such as [101], where a boosting approach including FuzzyKNN, evolutionary feature selection and decision trees is presented. Another example is [35], where a genetic algorithm is used to optimize a one-versus-all ensemble of FuzzyKNN classifiers. In [19], Chua et al. proposed an hybrid algorithm including a genetic fuzzy system, FuzzyKNN and a weighting scheme for the distance function of the classifier. Also, in [14], a model integrating FuzzyKNN and several multi layer neural networks as the core of a Mamdani type fuzzy inference system is proposed.

2.1.8. Applications

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

The first application was presented by Cabello et al. [13] where the Fuzzy C-Means clustering algorithm and FuzzyKNN are used together to tackle a problem of arrhythmia detection. Other recent applications are [17], where diabetes diseases are diagnosed by incorporating FuzzyKNN to a full artificial immune recognition system, and [71], where the joint use of particle swarm optimization, principal component analysis and FuzzyKNN is proposed for a thyroid disease diagnose problem.

Other medical technologies have also been benefited by the use of fuzzy nearest neighbor classifiers: Liao et al. ([66, 67, 68] presented several approaches for classifying radiographic images, including the use of feature extraction, Fuzzy C-Means clustering and FuzzyKNN. Another remarkable example is [64], where Leszczynski et al. analyze the performance of FuzzyKNN with different classic distance measures (euclidean, mahalanobis, etc.) in a framework of decision making in radiotherapy.

Another major field of application is bioinformatics. Many approaches for protein identification and prediction includes FuzzyKNN [47, 88, 44], with some of them incorporating additional mechanisms, such as [60], where a parallel implementation of FuzzyKNN is suggested.

Outside of the medical and bioinformatics fields there is also plenty of applications selecting FuzzyKNN as a suitable classifier (for example, [80] developing a wine classification system, [46] where FuzzyKNN is used to classify web documents, and [87] in which a computer vision approach for duck meat color classification is presented).

Moreover, it is also easy to find other applications in which FuzzyKNN is combined with preprocessing techniques (for example, [54] using Fuzzy C-Means for preprocessing data describing a cellular manufacturing system, or [63] which includes principal component analysis for reducing the dimensionality of data in a mold detection problem) or with other general methods (such as the recent proposals of [15] for bankruptcy prediction incorporating FuzzyKNN in a particle swarm optimization scheme, or [18] combining the output of several FuzzyKNN classifiers in a human action recognition problem.

Finally, there are not many applications including advanced fuzzy nearest neighbor classifiers, being [105] and [77] one of the most remarkable. In the former work, Zhu et al. presented a classifier inspired in D-SKNN, incorporating a fast implementation scheme, and applied it to image classification problems. In the later, Petridis et al. designed a k-NN method based on fuzzy interval numbers for the prediction of sugar production throughout different years.

2.2. Common properties of fuzzy nearest neighbor algorithms

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

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

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

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

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

- **Distance**: This category refers to the way in which the distances computed between the instance to be classified and each training instance are considered:

  - Inverse weight: The most common approach is to use the inverse of the similarity value computed (usually the Euclidean distance between the test and the training instance) as a weight for increasing the strength of the neighbor vote in the decision rule.
  - Distance modulation: Different schemes can be applied for incorporating the distances computed to the decision rule, modifiying its effect trough the use of additional procedures such as kernels or exponential relations.
  - Not used: Some of the techniques analyzed does not consider the absolute value of similarity computed in the decision rule. Still, they use distances to find the nearest neighbors of the test instance, but these values are disregarded as soon as the neighbors have been found.

- **Voting**: The definition of the voting rule used by the classifier. Typically, an additive scheme is chosen, which means that votes emitted by each neighboring instance (possibly weighted by its relative distance to the test instance) are added to create the final output of the classifier. However, different voting schemes may be selected:

  - Classical: An additive scheme is used for combining the neighbors' votes.
  - Global: Every instance in the training set is considered for the voting process (instead of just the neighboring instances).
  - Best neighbor: Only the best neighbor found among the k nearest ones (not necessarily the nearest) is used to determine the output of the classifier.

- **Others**: In this category, other relevant traits of nearest neighbor classifiers are included:

  - Independence of k: The method does not requires a value of k to be set for the decision rule.
  - Preprocessing: Besides the classification process, this technique also performs some kind of data preprocessing. Thus, as a side effect, the original training data is usually reduced (for example, by means of a prototype selection or generation technique). Note, however, that the main objective of the method is still the classification task.
  - Center based: The classification is oriented to relate test instances with the class whose center is nearest. This effect - desirable for many classification problems, although it may be harmful in certain cases - is typical for those techniques which rely on a clustering procedure for analyzing the training data.

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

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

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

Table 1: Common characteristics of fuzzy nearest neighbor algorithms

Acronym/Name	**Ref.**	**Member.**				**Distance**			**Voting**			**Others**		
		Crisp	Fuzzy	Posibilistic	Intuitionistic	Inverse weight	Distance modulation	Not used	Classical	Global	Best neighbor	Independence of k	Preprocessing	Center based
JFKNN	[55]		✓					✓		✓		✓		
FuzzyKNN	[58]		✓			✓			✓					
FuzzyNPC	[58]	✓				✓					✓	✓	✓	✓
FCMKNN	[9]		✓					✓	✓					✓
Kissiov et al.	[61]		✓					✓	✓					
D-SKNN	[25]			✓			✓		✓					
IFSKNN	[37]				✓			✓	✓					✓
IF-KNN	[62]				✓			✓	✓					
FENN	[100]		✓			✓			✓				✓	
VWFuzzyKNN	[39]		✓			✓			✓					
IFV-NP	[38]				✓		✓		✓			✓	✓	✓
Mitchell et al.	[72]	✓					✓			✓		✓		
IT2FKNN	[20]		✓			✓			✓					
PosIBL	[48]			✓			✓			✓		✓		
FRKNNA	[12]		✓					✓	✓					
Pham et al.	[78]		✓			✓			✓					
Gayar et al.	[34]		✓			✓					✓	✓	✓	✓
GAFuzzyKNN	[45]		✓			✓			✓					
FRNN	[83]			✓			✓			✓		✓		
PFKNN	[6]		✓			✓			✓				✓	
CFKNN	[104]		✓			✓			✓				✓	
FRNN-FRS	[52]	✓					✓				✓			
FRNN-VQRS	[52]	✓					✓				✓			
Qu et al.	[79]	✓					✓				✓			
Desterke et al.	[27]			✓			✓		✓					

3. Taxonomy

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

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

- For fuzzy sets based approaches, the main differential characteristics are the independence to the k value and the usage of distances to weight the computation of the votes in the decision rule.
- Possibilistic methods are also categorized depending if they are dependent to the set up of the k value or not.
- Similarly to the latter, intuitionistic fuzzy sets based methods are also categorized by their dependence to the set up of the k value. Besides, intuitionistic methods dependent to k can be further characterized as center based methods, focused on determining the center of each of the classes of the problem and adapt their classification scheme to the centers found.
- Fuzzy rough sets based method can be characterized regarding the scheme used for representing the membership of the training instances to the classes of the problem: The can use either fuzzy weights, possibilistic weights or crisp weights.
- The natural way of classifying preprocessing based approaches is to refer to the preprocessing field in which their are inspired (prototype selection or prototype generation). In addition, some prototype generation based techniques can be categorized as center based methods.

The properties displayed in this taxonomy may be very helpful in understanding how an specific algorithm works, enabling also the inclusion of new algorithms developed in the future. Albeit other different schemes could have

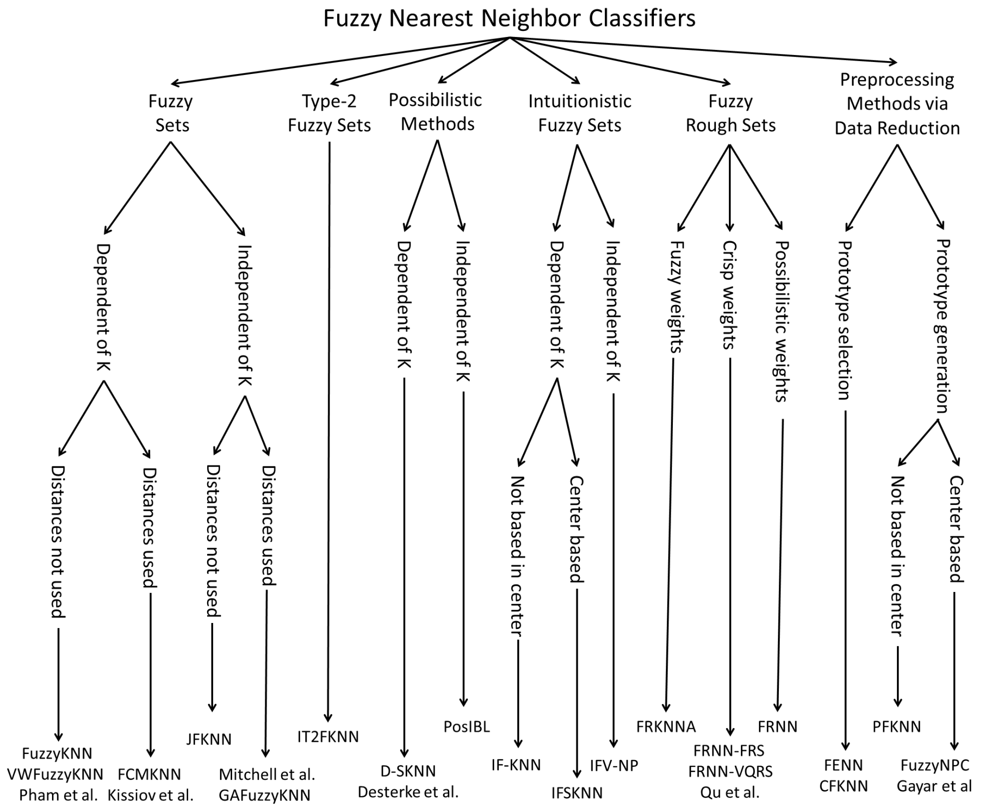


Figure 1: Proposed taxonomy of fuzzy nearest neighbor classifiers. Each method is be categorized into one of the six major classes: Fuzzy sets, type-2 fuzzy sets possibilistic methods, intuitionistic fuzzy sets, fuzzy rough sets or preprocessing approaches via data reduction. These classes are further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

been chosen here, the different levels established ensure that any technique (already analyzed or new) can be easy placed in one of the major categories, using the second and the rest of levels to refine its categorization as much as necessary.

4. Experimental framework for fuzzy nearest neighbor classifiers

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

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

The elements included in the framework are the following:

- Data sets: A large set of 44 well-known supervised classification data sets is provided, and their main characteristics are described.
- Fuzzy nearest neighbor classifiers: The framework features a library including the most relevant fuzzy nearest neighbor classifiers in the state of the art.
- Comparison algorithms: A collection of several representative crisp nearest neighbor classifiers is presented. They will be considered for testing the behavior of the best performing fuzzy nearest neighbor classifiers in a more general scenario.
- Parameters configuration: The guidelines followed to configure the parameter of each method are described. Also, the set up considered for the k parameter is discussed in depth, given its importance with respect to most of the methods.
- Performance measures: Several performance measures have been considered for analyzing the behavior of the methods. Their characteristics are described, as well as the motivations for including them into the experimental study.

- Statistical procedures: Several hypothesis testing procedures are considered to determine whether the differences found in the experimental study between the performance of multiple algorithms are significant or not.

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

4.1. Data sets

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

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

The data sets considered are partitioned by using the ten folds cross-validation (10-fcv) procedure, and their values are normalized in the interval $[0, 1]$ to equalize the influence of attributes with different range domains.

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

4.2. Fuzzy nearest neighbor classifiers

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

Table 3 list the methods included. For each one we provide its **acronym**, **name**, **year** of publication, and the main reference (**Ref.**) describing the work. We consider that this selection properly represents the current state of the art in the area. Note that we have not considered those approaches whose description in its original work was incomplete, or whose use require additional resources (such as [61], where expert human knowledge is required to be gathered prior to running the algorithm).

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

Table 2: Data sets included in the framework

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

Table 3: List of methods included in the framework

Acronym	**Name**	**Year**	**Ref.**
JFKNN	Jóźwik Fuzzy K-Nearest Neighbor algorithm	1983	[55]
FuzzyKNN	Fuzzy K-Nearest Neighbors classifier	1985	[58]
FuzzyNPC	Fuzzy Nearest Prototype classifier	1985	[58]
FCMKNN	Fuzzy C-Means K-Nearest Neighbors classifier	1986	[9]
D-SKNN	Dempster-Shafer theory based K-Nearest Neighbors classifier	1995	[25]
IFSKNN	Intuitionistic Fuzzy Sets K-Nearest Neighbors classifier	1995	[37]
IF-KNN	Intuitionistic Fuzzy K-Nearest Neighbors classifier	1995	[62]
FENN	Fuzzy Edited Nearest Neighbor classifier	1998	[100]
VWFuzzyKNN	Variance Weighted Fuzzy K-Nearest Neighbors classifier	1999	[39]
IFV-NP	Intuitionistic Fuzzy Version of K-Nearest Neighbors classifier	2000	[38]
IT2FKNN	Interval Type-2 Fuzzy K-Nearest Neighbors classifier	2003	[20]
PosIBL	Possibilistic Instance Based Learning	2003	[48]
FRKNNA	Fuzzy Rough K-Nearest Neighbors Approach	2003	[12]
GAFuzzyKNN	Genetic Algorithm for Fuzzy K-Nearest Neighbors classifier	2005	[45]
FRNN	Fuzzy-Rough Nearest Neighbor algorithm	2007	[83]
PFKNN	Pruned Fuzzy K-Nearest Neighbors classifier	2010	[6]
FRNN-FRS	Fuzzy-Rough Nearest Neighbor classifier - Fuzzy Rough Sets	2011	[52]
FRNN-VQRS	Fuzzy-Rough Nearest Neighbor classifier - Vaguely Quantified Rough Sets	2011	[52]
CFKNN	Condensed Fuzzy K-Nearest Neighbors classifier	2011	[104]

4.3. Comparison algorithms

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

- **k-NN classifier (k-NN):** The performance of the k-NN classifier will be studied as a reference for the rest of methods [23].
- **Edited Nearest Neighbors (ENN):** A prototype selection algorithm based on the edition of noisy instances. Instances whose class do not match their nearest neighbors' class are removed from the training set. After the edition process, the k-NN classifier is used to obtain the final classification [93].
- **Integrated Decremental Instance-Based Learning algorithm (IDIBL):** An integrated model featuring instance selection, selection of kernel function for the voting process, and automatic determination of the k value and other related parameters [94].
- **Adaptive k nearest neighbors classifier (KNNAdaptive):** A modification of the distance measure of the NN rule. Distances in this method are divided by the distance of the reference prototype to its nearest enemy (the nearest prototype from a different class) [90].
- **k Symmetrical nearest neighbors classifier (KSNN):** A modification of the voting rule of the NN classifier, where votes are considered for those instances for which the test instance would be one of its k nearest neighbors [74].
- **Nearest Subclass Classifier (NSC):** An application of the minimum variance clustering method to the generation of prototypes for the NN rule [89].
- **Prototype Weighting algorithm (PW):** A gradient descent based algorithm developed for computing prototype weights to minimize the *leave one out* error of the NN rule over the training set [76].

4.4. Parameters configuration

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

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

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

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

4.5. Performance measures

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

Accuracy is defined as the number of successful hits relative to the total number of classifications. It has been by far the most commonly used metric for assessing the performance of classifiers for years [95, 4]. Cohen's kappa [21] is an alternative to the accuracy rate, a method, known for decades, that compensates for random hits in the same way as the AUC measure [10]. Kappa can be computed using the following expression:

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

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

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

4.6. Statistical procedures

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

When using this framework, we recommend to consider the use of nonparametric statistical tests [84]. Their use is preferred over parametric ones when the initial conditions that guarantee the reliability of the parametric tests (independence, normality and homocedasticity) may not be satisfied, which is a common issue in many machine learning experimental set-ups [32, 31].

Several nonparametric procedures are suited for the aforementioned cases. Specificly, for multiple comparisons involving several procedures, we will consider the use of the Friedman test, together with a post-hoc procedure for analyzing families of hyphoteses interrelated, namely the Shaffer post-hoc procedure. This set of statistical methods will allow to contrast and confirm the results obtained in the experimental studies carried out [33].

More information about these tests and other statistical procedures specifically designed for use in the field of machine learning can be found at the

SCI2S thematic public website on Statistical Inference in Computational Intelligence and Data Mining (`http://sci2s.ugr.es/sicidm`).

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

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

- A first stage testing the performance of the fuzzy nearest neighbor classifiers over the full collection of data sets included in the framework.
- A second stage featuring a comparison between the best performing fuzzy nearest neighbor classifiers and a selection of state of the art crisp nearest neighbor classifiers.

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

An extended version of the definitions and results obtained in this experimental study is publicly available in `http://sci2s.ugr.es/fuzzyKNN/study.php`.

5.1. First stage: Comparison of fuzzy nearest neighbor classifiers

In this first stage of the study, we have considered all the fuzzy nearest neighbors classifiers implemented in the library (excepting JFKNN, since it is not able to tackle the largest data sets in a reasonable running time). Average accuracy and kappa results have been collected in two different ways:

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

- Secondly, the best average accuracy/kappa value per data set has been chosen (considering $k \in \{3, 5, 7, 9\}$ again). The average results obtained using the best value of k for each data set have been termed as **Accuracy/Kappa Best)** results.

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

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

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

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

- Considering accuracy results with a fixed value of k, the best algorithms are IT2FKNN, GAFuzzyKNN and FuzzyKNN. If the algorithms are compared considering their respective categories in the taxonomy, the best performing algorithms are FuzzyKNN (Fuzzy Sets), IT2FKNN (Type-2 Fuzzy Sets), D-SKNN (Possibilistic methods), IF-KNN (Intuitionistic Fuzzy Sets), FRNN-FRS (Fuzzy Rough Sets) and FENN (Preprocessing Methods via Data Reduction). It is also noticeable that low values for the k parameter (3 and 5) produce better results for most of the methods, excepting IT2FKNN and PFKNN.

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

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

Table 5: Summary results of Friedman & Shaffer tests for Accuracy (Fixed k, Stage 1)

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

- Considering accuracy results with the best value of k, most of the former conclusions holds. However, in this case GAFuzzyKNN achieves the best accuracy result, the differences between D-SKNN and the top 3 algorithms are lower. In general, all methods are benefited if the best value of k is chosen for each particular data set, although D-SKNN, IFSKNN and IFV-NP are the methods which obtain a greater benefit (more than 0.01 additional accuracy, on average).

- Considering the kappa performance measure with a fixed value of k, the best algorithms are still IT2FKNN, GAFuzzyKNN and FuzzyKNN. However, in this case GAFuzzyKNN is highlighted as the best method of the Fuzzy Sets family. Other noticeable differences are the relative improvement achieved by FRNN-FRS and FRNN-VQRS, and the performance drop suffered by FENN and PFKNN. Regarding the value of the k parameter, in this case medium values (5 and 7) are preferred generally by the best performing algorithms, excepting IF-KNN and FRNN-FRS.

- Considering the best value of k in the analysis with the kappa measure, D-SKNN can be also considered as the best algorithm (together with IT2FKNN, GAFuzzyKNN and FuzzyKNN). FENN achieves a better relative result and the relative performance of FRNN-VQRS is diminished. Again, all methods are benefited if the best value of k is chosen for each particular data set, but the greater improvement is obtained by D-SKNN (more than 0.02 additional kappa, on average).

- Finally, when running time is considered, the most noticeable result is the high computational cost of the GAFuzzyKNN and PFKNN methods (due to their wrapped based nature). The rest of methods are relatively cheap, computationally speaking, but PosIBL, FRNN-VQRS and FRNN-FRS, and D-SKNN obtain better results in this category. FuzzyNPC is, by far, the most efficient method. However this contrast with its poor results in all precision measures.

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

The results obtained enable us to give several suggestions and recommendations about the use of these fuzzy nearest neighbor, depending on the performance desired for a specific task:

- If very high accuracy is required, then GAFuzzyKNN, IT2FKNN or FuzzyKNN should be selected given their outstanding overall performance concerning this measure. However, the high computational cost of GAFuzzyKNN should also be considered if this technique is chosen. Other suitable options for high accuracy without a large running time are IF-KNN and D-SKNN. FENN could also be chosen as an accurate method with the added feature of the removal of noisy instances from the training set, which should also help in reducing the running time in the final classification phase.

- There are very few differences if kappa is considered instead of accuracy. FRNN-FRS shows a small improvement in its results, which suggests that it keeps a better balance (when compared with other methods with similar performance) over the classes of the problems, without biasing towards the majority classes. Apart from that, the lack of differences between accuracy and kappa suggest that the best performing algorithms are not biased toward obtaining a good precision in the majority classes of the problems, thus balancing their efforts over all the classes of the domains considered.

- D-SKNN should be the technique to select if a good performance is required without consuming too much computational resources. It is one of the fastest methods analyzed in the study and is not outperformed by any other fuzzy nearest neighbor method, thus becoming a fast and reliable choice in these kind of situations. A second recommendation would be FRNN-FRS, which is slightly faster than FRNN-FRS and still keeps good precision rates.

5.2. Second stage: Comparison with crisp nearest neighbor approaches

In the second stage of the study, a comparison including the best performing fuzzy nearest neighbor classifiers and several crisp nearest neighbor

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

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

Table 7: Summary results of Friedman & Shaffer tests for Accuracy (Fixed k, Stage 2)

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

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

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

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

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

- If accuracy with a fixed value of k is considered, the five best positions are achieved by fuzzy nearest neighbor classifiers (GAFuzzyKNN, IT2FKNN, FuzzyKNN, D-SKNN and IF-KNN). None of the 7 fuzzy nearest neighbor classifiers included shows a performance lower than the original k-NN rule. It is also interesting to note the improvement achieved by FuzzyKNN and FENN over their direct crisp counterparts, KNN and ENN (an improvement of 0.0295 and 0.0142, respectively).

- The former results holds, in general, if the best value of k is considered individually. Also, in this case, the crisp methods KSNN and NSC shows a performance comparable with some of the best fuzzy nearest neighbor classifiers. FRNN-FRS is the only fuzzy nearest neighbor classifiers whose performance drops below KNN under these conditions.

- The results obtained using the kappa performance measure with a fixed value of k still highlights GAFuzzyKNN, IT2FKNN and FuzzyKNN as the better algorithms. On a second level, KSNN and FRNN-FRS shows an improvement, being comparable to D-SKNN and IF-KNN in this category. In this case, only FENN's kappa falls below KNN's.

- Considering the best value of k with the kappa measure, the differences among methods are more tight. Most of the former results are the same, being the most noticeable differences the improvement achieved by NSC and the relative drop of the position of FRNN-FRS (which almost does not get any improvement by allowing to set the best value of k in each data set).

- Finally, there are not many differences between crisp and fuzzy methods with respect to running time. Both families have very fast methods (FRNN-FRS, D-SKNN, KNN, NSC and ENN) and slower ones (IDIBL and GAFuzzyKNN), which shows that there is not an additional computational cost of fuzzy mechanisms are introduced to improve the nearest neighbor rule, in comparison with crisp based mechanisms.

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

In general, this second study has shown that fuzzy nearest neighbors classifiers can demonstrate a positive performance if considered within the state of the art in nearest neighbor classification: Several methods offer better precision (accuracy and kappa rates) and they are not specially slower or faster than the crisp approaches. Hence, they are a suitable option for standard supervised learning tasks, in which high accuracy at a relatively lower computational cost is required.

6. Future prospects

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

Besides, the conclusions gathered along the survey can be used to suggest some unaddressed challenges and remarks which could be very valuable for a further development of the field:

- Most of the techniques reviewed are not able to dealt properly with nominal (categorical) attributes and missing values, or directly does not describe a method for manage them. Although some solutions can be incorporated from the classical classification field (such as advanced similarity measures for tackling nominal attributes [92] or imputation techniques for handling missing data [70]), there is still the necessity of a specific solution which enable fuzzy nearest neighbor classifiers handle these kind of data with ease.

- A key aspect in the performance of the fuzzy nearest neighbor algorithms is the way in which the membership values to the classes are computed. It is true that there is a rich variety of way for representing these values. However, most of them are based on the concept of locality (that is, membership are assigned in accordance with the nearest instances in the training data). Other schemes of analysis, based in different concepts such as global characteristics of the data could be incorporated to develop new membership assignation schemes, prone to improve further the generalization capabilities of the algorithms. The development of a new class of preprocessing techniques could also be helpful here, if they are applied as a way of refining an initial configuration of memberships for a data set.

- Using the theoretical developments shown in [97] as a starting point, new voting schemes could be designed (probably in an automatic way), far from the traditional majority rules or the search for a best single instance. The definition of *ad hoc* voting rules, specific to the current problem tackled by the classifier, would enable a specialized treatment of the intrinsic characteristics of the data. These rules would allow fitting the classifier to the problem addressed, further enhancing its classification performance.

7. Conclusions

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

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

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

Acknowledgements

Supported by the Projects TIN2011-28488, P10-TIC-06858 and P11-TIC-9704. J. Derrac holds an FPU scholarship from the Spanish Ministry of Education.

References

[1] D.W. Aha, D. Kibler, M.K. Albert, Instance-based learning algorithms, Machine Learning 6 (1991) 37–66.

[2] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera, Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework, Journal of Multiple-Valued Logic and Soft Computing 17 (2011).

[3] J. Alcalá-Fdez, L. Sánchez, S. García, M.J. del Jesus, S. Ventura, J.M. Garrell, J. Otero, C. Romero, J. Bacardit, V.M. Rivas, J.C. Fernández, F. Herrera, KEEL: a software tool to assess evolutionary algorithms for data mining problems, Soft Computing 13 (2008) 307–318.

[4] E. Alpaydin, Introduction to Machine Learning, The MIT Press, 2 edition, 2010.

[5] A. Andoni, P. Indyk, Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions, Communications of the ACM 51 (2008).

[6] M. Arif, M.U. Akram, F.A. Afsar, F.A.A. Minhas, Pruned fuzzy k-nearest neighbor classifier for beat classification, Journal of Biomedical Science and Engineering 3 (2010).

[7] K. Atanassov, Intuitionistic fuzzy sets, Fuzzy Sets and Systems 20 (1986) 87–96.

[8] J. Bedzek, P. Castelaz, Prototype classification and feature selection with fuzzy sets, IEEE Transactions on Systems, Man, and Cybernetics 7 (1977) 87–92.

[9] J. Bedzek, S.K. Chuah, Generalized k-nearest neighbor rules, Fuzzy Sets and Systems 18 (1986) 237–256.

[10] A. Ben-David, A lot of randomness is hiding in accuracy, Engineering Applications of Artificial Intelligence 20 (2007) 875–885.

[11] M. Béreau, B. Dubuisson, A fuzzy extended k-nearest neighbors rule, Fuzzy Sets and Systems 44 (1991) 17–32.

[12] H. Bian, L. Mazlack, Fuzzy-rough nearest neighbor classification approach, in: Proceedings of the 22th International Conference of the North American Fuzzy Information Processing Society (NAFIPS'03),Chicago, Illinois, USA, July 24-26, pp. 500–505.

[13] D. Cabello, S. Barro, J. Salceda, R. Ruiz, J. Mira, Fuzzy k-nearest neighbor classifiers for ventricular arrhytmia detection, International Journal of Biomedical Computing 27 (1991) 77–93.

[14] O. Castillo, P. Melin, E. Ramírez, J. Soria, Hybrid intelligent system for cardiac arrhythmia classification with fuzzy k-nearest neighbors and neural networks combined with a fuzzy system, Expert Systems with Applications 39 (2012) 2947–2955.

[15] H.L. Chen, B. Yang, G. Wang, J. Liu, X. Xu, S. Wang, D. Liu, A novel bankruptcy prediction model based on an adaptive fuzzy k-nearest neighbor method, Knowledge Based Systems 24 (2011) 1348–1359.

[16] Y. Chen, E.K. Garcia, M.R. Gupta, A. Rahimi, L. Cazzanti, Similarity-based classification: Concepts and algorithms, Journal of Machine Learning Research 10 (2009) 747–776.

[17] M.A. Chikh, M. Saidi, N. Settouti, Diagnosis of diabetes diseases using an artificial immune recognition system2 (AIRS2) with fuzzy k-nearest neighbor, Journal of medical systems 36 (2012) 2721–2729.

[18] T.W. Chua, K. Leman, N.T. Pham, Human action recognition via sum-rule fusion of fuzzy k-nearest neighbor classifiers, in: IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2011), Taipei, Taiwan, June 27-30, pp. 484–489.

[19] T.W. Chua, W.W. Tan, A new fuzzy rule-based initialization method for k-nearest neighbor classifier, in: Proceedings of the IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'09), Jeju island, Korea, August 20-24, pp. 415–420.

[20] F. Chung-Hoon, C. Hwang, An interval type-2 fuzzy k-nearest neighbor, in: Proceedings of the 12th IEEE International Conference on Fuzzy Systems (IEEE FUZZY'03), St. Louis, Missouri, USA, May 25-28, pp. 802–807.

[21] J. Cohen, A coefficient of agreement for nominal scales, Educational and Psychological Measurement 20 (1960) 37–46.

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

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

[24] M. De Cock, C. Cornelis, E.E. Kerre, Fuzzy rough sets: The forgotten step, IEEE Transactions on Fuzzy Systems 15 (2007) 121–130.

[25] T. Denoeux, A k-nearest neighbor classification rule based on dempster-shafer theory, IEEE Transactions on Systems, Man, and Cybernetics 25 (1995) 804–813.

[26] J. Derrac, C. Cornelis, S. Garca, F. Herrera, Enhancing evolutionary instance selection algorithms by means of fuzzy rough set based feature selection, Information Sciences 186 (2012) 73–92.

[27] S. Destercke, A k-nearest neighbours method based on imprecise probabilities, Soft Computing 16 (2012) 833–844.

[28] E. Fix, J. Hodges, Discriminatory analysis, nonparametric discrimination: Consistency properties, Technical Report 4, USAF School of Aviation Medicine, 1951.

[29] A. Frank, A. Asuncion, UCI machine learning repository, 2010.

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

[31] S. García, A. Fernández, J. Luengo, F. Herrera, A study of statistical techniques and performance measures for genetics-based machine learning: Accuracy and interpretability, Soft Computing 13 (2009) 959–977.

[32] S. García, A. Fernández, J. Luengo, F. Herrera, Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power, Information Sciences 180 (2010) 2044–2064.

[33] 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.

[34] N. Gayar, F. Schwenker, G. Palm, A study of the robustness of knn classifiers trained using soft labels, in: Proceedings of the Second IAPR Workshop on Artificial Neural Networks in Pattern Recognition (ANNPR 2006), Ulm, Germany, August 31 - September 2, pp. 67–80.

[35] V. Gesu, G. Bosco, Combining one class fuzzy knn's, in: 7th International Workshop on Fuzzy Logic and Applications (WILF 2007), Camogli, Italy, July 7-10, pp. 152–160.

[36] P. Grother, G.T. Candela, J.L. Blue, Fast implementations of nearest-neighbor classifiers, Pattern Recognition 30 (1997) 459–465.

[37] S. Hadjitodorov, An intuitionistic fuzzy sets application to the k-nn method, Notes on Intuitionistic Fuzzy Sets 1 (1995) 66–69.

[38] S. Hadjitodorov, An intuitionistic fuzzy version of the nearest prototype classification method, based on a moving-of-pattern procedure, International Journal of General Systems 30 (2000) 155–165.

[39] J.H. Han, Y.K. Kim, A fuzzy k-nn algorithm using weights from the variance of membership values, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (IEEE CVPR'99), Ft. Collins, Colorado, USA, June 23-25, pp. 394–399.

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

[41] L. Hartert, M.S. Mouchaweh, P. Billaudel, A semi-supervised dynamic version of fuzzy k-nearest neighbours to monitor evolving systems, Evolving Systems 1 (2010) 3–15.

[42] T. Hastie, R. Tibshirani, Discriminant adaptive nearest neighbor classification, IEEE Transactions on Pattern Analysis and Machine Intelligence 18 (1996) 607–616.

[43] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning: Data Mining, Inference and Prediction, Springer, New York Berlin Heidelberg, 2 edition, 2009.

[44] M. Hayat, A. Khan, Discriminating outer membrane proteins with fuzzy k-nearest neighbor algorithms based on the general form of chou's PseAAC, Protein and Peptide Letters 19 (2012) 411–421.

[45] X. Hu, C. Xie, Improving fuzzy k-nn by using genetic algorithm, Journal of Computational Information Systems 1 (2005) 203–213.

[46] Z. Huan, N. Yi, H. Nie, Web document classification based on fuzzy k-nn algorithm, in: Proceedings of the 2009 International Conference on Computational Intelligence and Security (CIS 2009), Beijing, China, December 11-14, pp. 193–196.

[47] Y. Huang, Y. Li, Prediction of protein subcellular locations using fuzzy k-nn method, Bioinformatics 20 (2004) 21–28.

[48] E. Hüllermeier, Possibilistic instance-based learning, Artificial Intelligence 148 (2003) 335–383.

[49] H. Ishibuchi, T. Nakashima, Evolution of fuzzy nearest neighbor neural networks, in: Proceedings of the 1997 IEEE International Conference on Evolutionary Computation (IEEE CEC'97), Indianapolis, USA, April 13-16, pp. 673–678.

[50] M.Z. Jahromi, E. Parvinnia, R. John, A method of learning weighted similarity function to improve the performance of nearest neighbor, Information Sciences 179 (2009) 2964–2973.

[51] R. Jensen, C. Cornelis, Fuzzy-rough instance selection, in: IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2010), Barcelona, Spain, July 18-23, pp. 1776–1782.

[52] R. Jensen, C. Cornelis, Fuzzy-rough nearest neighbour classification, Transactions on Rough Sets 13 (2011) 56–72.

[53] J.Y. Jiang, S.C. Tsai, S.J. Lee, Fsknn: Multi-label text categorization based on fuzzy similarity and k nearest neighbors, Expert Systems with Applications 39 (2012) 2813–2821.

[54] K. Josien, T.W. Liao, Integrated use of fuzzy c-means and fuzzy knn for gt part family and machine cell formation, International Journal of Production Research 38 (2000) 3513–3536.

[55] A. Jóẃik, A learning scheme for a fuzzy k-nn rule, Pattern Recognition Letters 1 (1983) 287–289.

[56] X.M. Kang, X.P. Liu, J.H. Zhai, M.Y. Zhai, Instances selection for nn with fuzzy rough technique, in: Proceedings of the International Conference on Machine Learning and Cybernetics (ICMLC 2011), Guilin, China, July 10-13, pp. 1097–1100.

[57] N. Karnik, J. Mendel, Q. Liang, Type-2 fuzzy logic systems, IEEE Transactions on Fuzzy Systems 7 (1999) 643–658.

[58] J.M. Keller, M.R. Gray, J.A. Givens, A fuzzy k-nearest neighbor algorithm, IEEE Transactions on Systems, Man, and Cybernetics 15 (1985) 580–585.

[59] A. K.Ghosh, On optimum choice of k in nearest neighbor classification, Computational Statistics & Data Analysis 50 (2006) 3113–3123.

[60] S. Kim, J. Sim, J. Lee, Fuzzy k-nearest neighbor method for protein secondary structure prediction and its parallel implementation, in: Proceedings of the International Conference on Intelligent Computing (ICIC 2006), Kunming, China, August 16-19, pp. 444–463.

[61] V. Kissiov, S. Hadjitodorov, A fuzzy version of the k-nn method, Fuzzy Sets and Systems 49 (1992) 323–329.

[62] L.I. Kuncheva, An intuitionistic fuzzy k-nearest neighbors rule, Notes on Intuitionistic Fuzzy Sets 1 (1995) 56–60.

[63] M. Kuske, R. Rubio, A.C. Romain, J. Nicolas, S. Marco, Fuzzy k-nn applied to moulds detection, Sensors and Actuators B 106 (2004) 52–60.

[64] K. Leszczynski, S. Cosby, R. Bissett, D. Provost, S. Boyko, S. Loose, E. Mvilongo, Application of a fuzzy pattern classifier to decision making in portal verification of radiotherapy, Physics in Medicine and Biology 44 (1999) 253–269.

[65] S. Liang, C. Li, A fast and scalable fuzzy-rough nearest neighbor algorithm, in: Proceedings of the IEEE Global Congress on Intelligent Systems (GCIS'09), Xiamen, China, May 19-21, pp. 311–314.

[66] T. Liao, D. Li, Two manufacturing applications of the fuzzy k-nn algorithm, Fuzzy Sets and Systems 92 (1997) 289–303.

[67] T. Liao, D. Li, Y. Li, Detection of welding flaws from radiographic images with fuzzy clustering methods, Fuzzy Sets and Systems 108 (1999) 145–158.

[68] T. Liao, D. Li, Y. Li, Extraction of welds from radiographic images using fuzzy classifiers, Information Sciences 126 (2000) 21–40.

[69] F.O.S.S. Lisboa, M. Nicoletti, A. Ramer, A version of the nge model suitable for fuzzy domains, Journal of Intelligent & Fuzzy Systems 18 (2007) 1–17.

[70] R.J. Little, D. Rubin, Statistical analysis with Missing Data, John Wiley and Sons, 2 edition, 2002.

[71] D.Y. Liu, H.L. Chen, B. Yang, X.E. Lv, L.N. Li, J. Liu, Design of an enhanced fuzzy k-nearest neighbor classifier based computer aided diagnostic system for thyroid disease, Journal of medical systems 36 (2012) 3243–3254.

[72] H.B. Mitchell, P.A. Schaefer, A soft k-nearest neighbor voting scheme, International Journal of Intelligent Systems 16 (2001) 459–468.

[73] T. Nakashima, H. Ishibuchi, Learning of fuzzy reference sets in nearest neighbor classification, in: Proceedings of the 18th International Conference of the North American Fuzzy Information Processing Society (NAFIPS'99), New York, USA, June 10-12, pp. 357–360.

[74] R. Nock, M. Sebban, D. Bernard, A simple locally adaptive nearest neighbor rule with application to pollution forecasting, International Journal of Pattern Recognition and Artificial Intelligence 17 (2003) 1369–1382.

[75] R. Ostermark, A fuzzy vector valued knn-algorithm for automatic outlaier detection, Applied Soft Computing 9 (2009) 1263–1272.

[76] R. Paredes, E. Vidal, Learning weighted metrics to minimize nearest-neighbor classification error, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (2006) 1100–1110.

[77] V. Petridis, V.G. Kaburlasos, Finknn: A fuzzy interval number k-nearest neighbor classifier for prediction of sugar production from populations of samples, Journal of Machine Learning Research 4 (2004) 17–37.

[78] T.D. Pham, An optimally wieghted fuzzy k-nn algorithm, in: Proceedings of the Third International Conference on Advances in Pattern Recognition, Part I (ICAPR 2005), Bath, UK, August 22-25, pp. 239–247.

[79] Y. Qu, C. Shang, Q. Shen, N.M. Parthaláin, W. Wu, Kernel-based fuzzy-rough nearest neighbour classification, in: IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2011), Taipei, Taiwan, June 27-30, pp. 1523–1529.

[80] C. Raptis, C. Siettos, C. Kiranoudis, G. Bafas, Classification of aged wine distillates using fuzzy and neural network systems, Journal of Food Engineering 46 (2000) 267–275.

[81] S. Roh, T. Ahn, W. Pedrycz, The refinement of models with the aid of the fuzzy k-nearest neighbors approach, IEEE Transactions on Instrumentation and Measurement 59 (2010) 604–615.

[82] J.L.A. Rosa, N.F.F. Ebecken, Data mining for data classification based on the knn-fuzzy method supported by genetic algorithm, in: Proceedings of the 5th international conference on high performance computing for computational science (VECPAR'02), Porto, Portugal, June 26-28. Lecture Notes in Computer Science, 2565, pp. 126–133.

[83] M. Sarkar, Fuzzy-rough nearest neighbor algorithms in classification, Fuzzy Sets and Systems 158 (2012) 2134–2152.

[84] D.J. Sheskin, Handbook of Parametric and Nonparametric Statistical Procedures, Chapman & Hall/CRC, 5 edition, 2011.

[85] S. Singh, A single nearest neighbour fuzzy approach for pattern recognition, International Journal of Pattern Recognition and Artificial Intelligence 13 (1998) 49–54.

[86] I. Triguero, J. Derrac, S. García, F. Herrera, A taxonomy and experimental study on prototype generation for nearest neighbor classification, IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 42 (2012) 86–100.

[87] D.C. Tu, J.H. Zhao, M.H. Liu, J. Shen, F. Yu, Preliminary study on quantification of duck color based on fuzzy k nearest neighbor method, Applied Mechanics and Materials 39 (2010) 210–215.

[88] T.Q. Tung, D. Lee, A method to improve protein subcellular localization prediction by integrating various biological data sources, BMC Bioinformatics 10 (2009) 1–10.

[89] C.J. Veenman, M.J.T. Reinders, The nearest subclass classifier: A compromise between the nearest mean and nearest neighbor classifier, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (2005) 1417–1429.

[90] J. Wang, P. Neskovic, L. Cooper, Improving nearest neighbor rule with a simple adaptative distance measure, Pattern Recognition Letters 28 (2007) 207–213.

[91] D. Wettschereck, D.W. Aha, T. Mohri, A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms, Artificial Intelligence Review 11 (1997) 273–314.

[92] D. Wilson, T. Martinez, Improved heterogeneous distance functions, Journal of Artificial Intelligence Research 6 (1997) 1–34.

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

[94] D.R. Wilson, T.R. Martinez, An integrated instance-based learning algorithm, Computational Intelligence 16 (2000) 1–28.

[95] I.H. Witten, E. Frank, M.A. Hall, Data Mining: Practical Machine Learning Tools and Techniques, Morgan Kaufmann Series in Data Management Systems, Morgan Kaufmann, 3 edition, 2011.

[96] X. Wu, V. Kumar (Eds.), The Top Ten Algorithms in Data Mining, Data Mining and Knowledge Discovery, Chapman & Hall/CRC, 2009.

[97] R.R. Yager, Using fuzzy methods to model nearest neighbor rules, IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics 32 (2002) 512–525.

[98] M. Yang, C. Chen, On strong consistency of the fuzzy generalized nearest neighbor rule, Fuzzy Sets and Systems 60 (1993) 273–281.

[99] M. Yang, C. Chen, Convergence rate of the fuzzy generalized nearest neighbor rule, Computers & Mathematics with Applications 27 (1994) 1–8.

[100] M. Yang, C. Cheng, On the edited fuzzy k-nearest neighbor rule, IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics 28 (1998) 461–466.

[101] S. Yu, S. Backer, P. Scheunders, Genetic feature selection combined with composite fuzzy nearest neighbor classifiers for hyperspectral satellite imagery, Pattern Recognition Letters 23 (2002) 183–190.

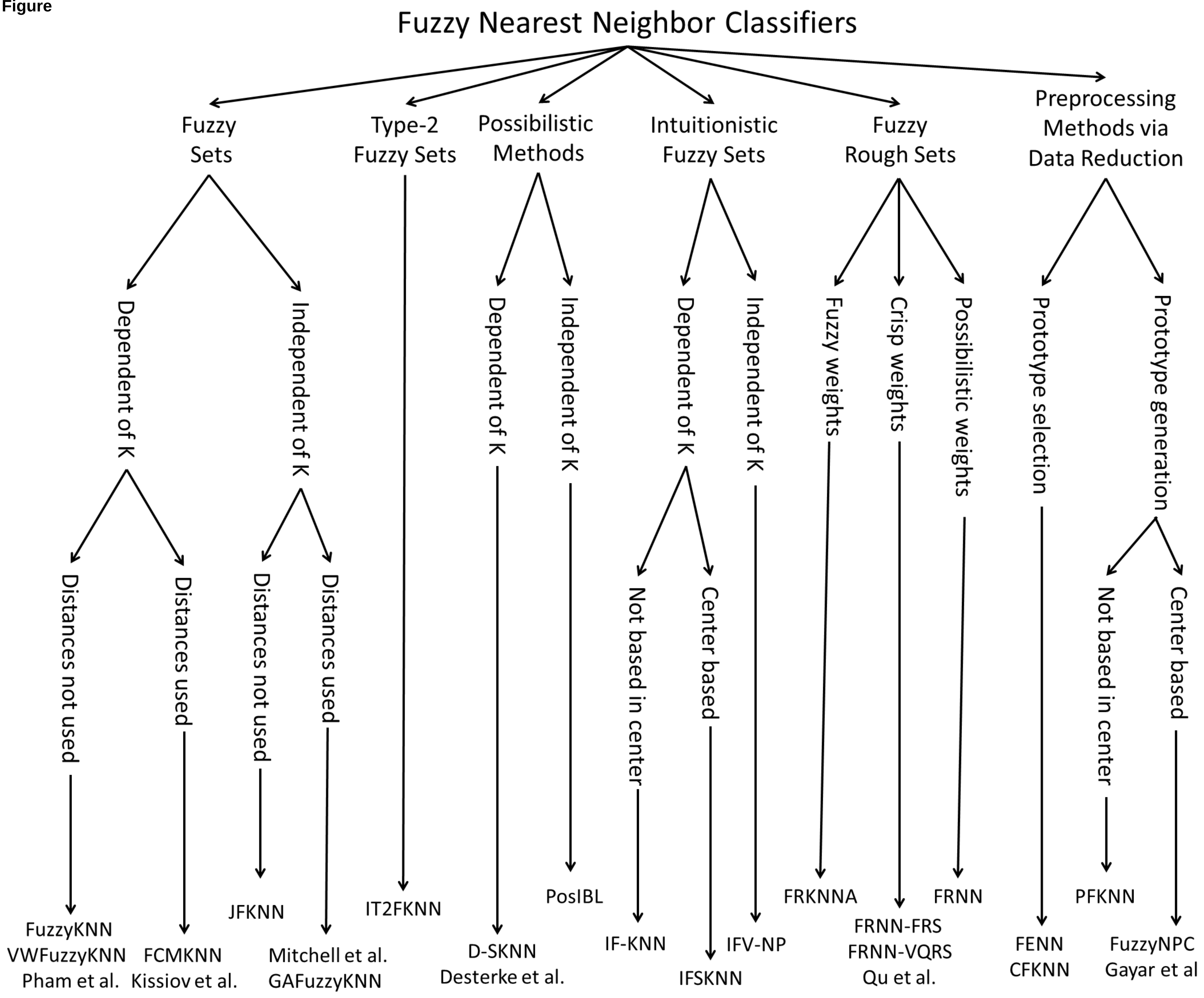
[102] L. Zadeh, Fuzzy sets, Information and Control 8 (1965) 338–353.

[103] L. Zadeh, Fuzzy sets as a basis for a theory of possibility, Fuzzy Sets and Systems 1 (1978) 3–28.

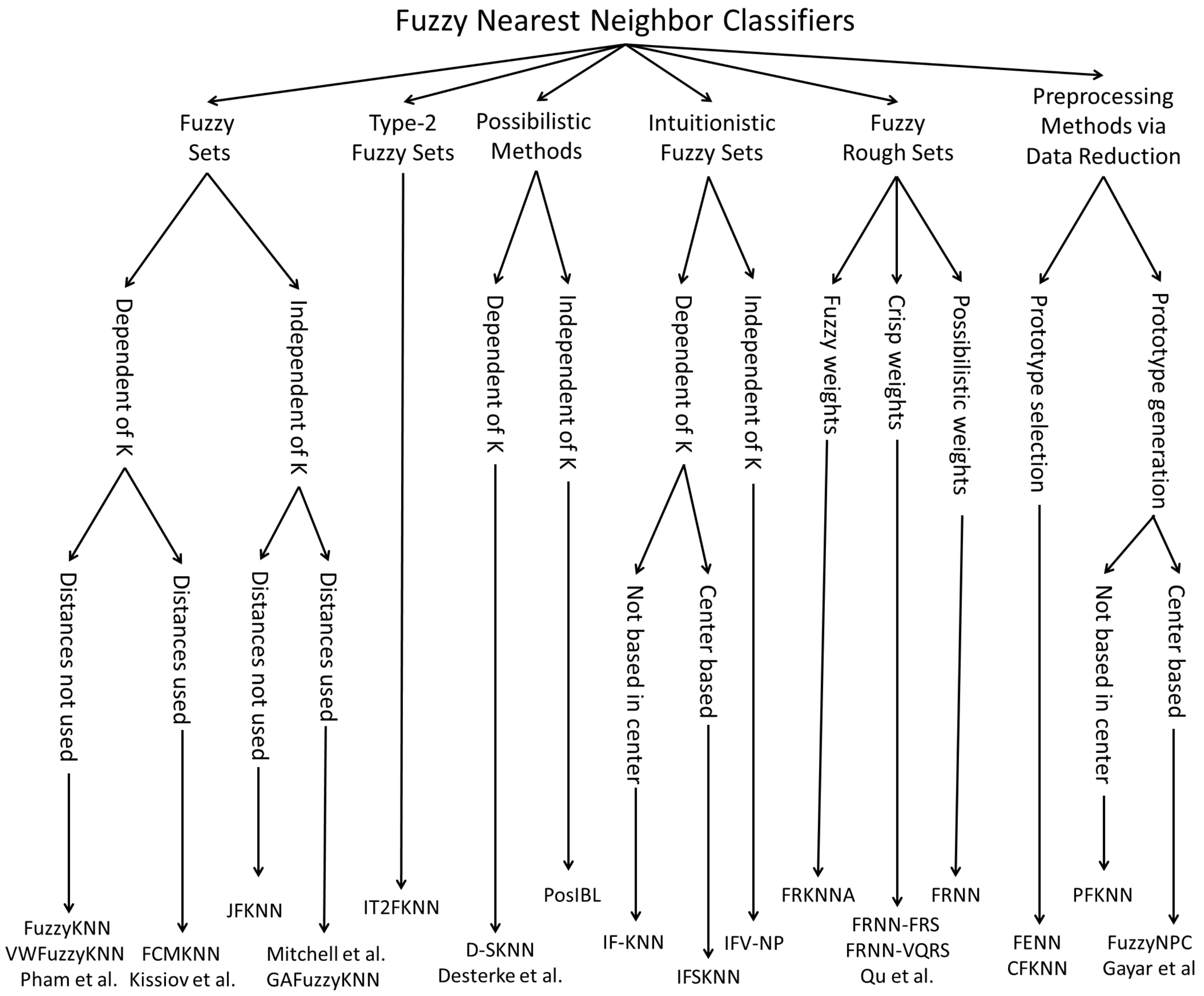
[104] J.H. Zhai, N. Li, M.Y. Zhai, The condensed fuzzy k-nearest neighbor rule based on sample fuzzy entropy, in: Proceedings of the 2011 International Conference on Machine Learning and Cybernetics (ICMLC'11), Guilin, China, July 10-13, pp. 282–286.

[105] H. Zhu, O. Basir, A k-nn associated fuzzy evidential reasoning classifier with adaptive neighbor selection, in: Proceedings of the Third IEEE International Conference on Data Mining (ICDM 2003), Melbourne, Florida, USA, December 19-22, pp. 709–712.

Figure


Fuzzy Nearest Neighbor Classifiers
Fuzzy Sets
Type-2 Fuzzy Sets
Possibilistic Methods
Intuitionistic Fuzzy Sets
Fuzzy Rough Sets
Preprocessing Methods via Data Reduction
Dependent of K
Independent of K
Dependent of K
Independent of K
Dependent of K
Independent of K
Fuzzy weights
Crisp weights
Possibilistic weights
Prototype selection
Prototype generation
Distances not used
Distances used
Distances not used
Distances used
Not based in center
Center based
Not based in center
Center based
FuzzyKNN
VWFuzzyKNN
Pham et al.
FCMKNN
Kissiov et al.
JFKNN
Mitchell et al.
GAFuzzyKNN
IT2FKNN
D-SKNN
Desterke et al.
PosIBL
IF-KNN
IFSKNN
IFV-NP
FRKNNA
FRNN-FRS
FRNN-VQRS
Qu et al.
FRNN
FENN
CFKNN
PFKNN
FuzzyNPC
Gayar et al

Figure


Fuzzy Nearest Neighbor Classifiers
Fuzzy Sets
Type-2 Fuzzy Sets
Possibilistic Methods
Intuitionistic Fuzzy Sets
Fuzzy Rough Sets
Preprocessing Methods via Data Reduction
Dependent of K
Independent of K
Distances not used
Distances used
Distances not used
Distances used
Dependent of K
Independent of K
Dependent of K
Independent of K
Not based in center
Center based
Fuzzy weights
Crisp weights
Possibilistic weights
Prototype selection
Prototype generation
Not based in center
Center based
FuzzyKNN
VWFuzzyKNN
Pham et al.
FCMKNN
Kissiov et al.
JFKNN
Mitchell et al.
GAFuzzyKNN
IT2FKNN
D-SKNN
Desterke et al.
PosIBL
IF-KNN
IFSKNN
IFV-NP
FRKNNA
FRNN-FRS
FRNN-VQRS
Qu et al.
FRNN
FENN
CFKNN
PFKNN
FuzzyNPC
Gayar et al

4.2 EIVF-kNN: An Evolutionary Interval Valued Fuzzy K-Nearest Neighbors Classifier

- J. Derrac, F. Chiclana, S. García, F. Herrera, EIVF-kNN: An Evolutionary Interval Valued Fuzzy K-Nearest Neighbors Classifier. **Submitted to IEEE Transactions on Fuzzy Sets and Systems**.
    - Status: **Submitted**.
    - Impact Factor (JCR 2011): 4.260.
    - Subject Category: Computer Science, Artificial Intelligence. Ranking 5 / 111 (**Q1**).
    - Subject Category: Engineering, Electrical & Electronic. Ranking 8 / 245 (**Q1**).

EIVF-kNN: An Evolutionary Interval Valued Fuzzy K-Nearest Neighbors Classifier

Journal:	*Transactions on Fuzzy Systems*
Manuscript ID:	Draft
Manuscript Type:	Full Papers
Date Submitted by the Author:	n/a
Complete List of Authors:	Derrac, Joaquín; University of Granada, Computer Science and A.I Chiclana, Francisco; De Montfort University, Division of Computer Engineering García, Salvador ; University of Jaén, Computer Science Herrera, Francisco; University of Granada, Computer Science and Artificial Intelligence;
Keywords:	Evolutionary Algorithms, Supervised Learning, Classification, Interval Valued Fuzzy Sets, Fuzzy Nearest Neighbor

EIVF-kNN: An Evolutionary Interval Valued Fuzzy K-Nearest Neighbors Classifier

Joaquín Derrac, Francisco Chiclana, Salvador García and Francisco Herrera, *Member, IEEE,*

Abstract

The K-Nearest Neighbors classifier has become a well-known, successful method for pattern classification tasks. In recent years, many enhancements to the original algorithm have been proposed. Fuzzy sets theory has been the basis of several proposed models towards the enhancement of the nearest neighbors rule, with the Fuzzy K-Nearest Neighbors (Fuzzy-kNN) classifier the most notable procedure in the field.

In this work we present a new nearest neighbor classifier, IVF-kNN, based on the use of type-2 interval valued fuzzy sets. The use and implementation of interval values facilitates the membership of the instances and the computation of the votes in a more flexible way than the original Fuzzy-kNN method, thus improving its adaptability to different supervised learning problems. An evolutionary version of our approach, EIVF-kNN, is also proposed, further improving the capabilities of the interval valued model through the adjustment of several key parameters of the original method. An experimental study, contrasted by the application of nonparametric statistical procedures, is carried out to compare the performances of these and some advanced fuzzy nearest neighbor classifiers, over several well-known classification problems. The results enable us to safely confirm that EIVF-kNN has the qualities necessary to become a strong approach in the field of fuzzy nearest neighbor classification.

Index Terms

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

F. Chiclana is with the De Montfort University Interdisciplinary Group in Intelligent Transport Systems (DIGITS), Faculty of Technology, De Montfort University, Leicester LE1 9BH, United Kingdom (e-mail: chiclana@dmu.ac.uk).

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

Fuzzy Nearest Neighbor, Interval Valued Fuzzy Sets, Evolutionary Algorithms, Supervised Learning, Classification.

I. Introduction

The k Nearest Neighbors classifier (kNN) [1] is one of the most popular supervised learning methods. It is a nonparametric method which does not rely on building a model during the training phase, and whose classification rule is based on a given similarity function between the training instances and the test instance to be classified. Since its definition, k-NN has become one of most relevant algorithms in data mining [2], and is an integral part of many applications of machine learning in various domains [3], [4].

In nearest neighbor classification, fuzzy sets can be used to model the degree of membership of each instance to the classes of the problem. This approach, known as the Fuzzy K-Nearest Neighbor (Fuzzy-kNN) classifier [5], has been shown to be an effective improvement of kNN.

This fuzzy approach overcomes the drawback associated with the kNN classifier, in which equal importance is given to every instance in the decision rule, regardless of its typicalness as a class prototype and the distance between it and the pattern to be classified. Fuzzy memberships enable Fuzzy-kNN to achieve higher accuracy rates in most classification problems. This is also the reason why it has been the preferred choice in several applications in medicine [6], [7], economy [8], bioinformatics [9], industry [10] and many other fields.

Thus, the definition of fuzzy memberships is a fundamental aspect of Fuzzy-kNN. Although they can be set through expert knowledge, or by analyzing local data around each instance (as in [5]), there may be still a lack of knowledge associated with the assignation of a single value to the membership. To overcome this difficulty, interval valued fuzzy sets (IVFSs) [11], a particular case of type-2 fuzzy sets [12], may be used.

IVFSs allow membership values to defined by using a lower and an upper bound. The interval based definition includes not only a greater degree of flexibility than just using a single value, but also enables us to measure the degree of ignorance with the length of the interval [13], [14]. Following this approach, IVFSs have been successfully applied in the development of fuzzy systems for classification [15]–[17]. In the case of nearest neighbor classification, this enables the representation of the uncertainty associated with the true class (or classes) to which every instance belongs, in the context of most standard, supervised classification problems.

In this paper, we present an Evolutionary Interval Valued Fuzzy k-Nearest Neighbors classifier (EIVF-kNN). Its development is composed of two stages, the first one extending the Fuzzy-kNN classifier through the incorporation of IVFS, and the second including an evolutionary optimization process to tune several key parameters of the model:

- In the first stage, we extend the Fuzzy-kNN classifier via the implementation of interval values to represent the membership of each instance to the classes and the votes cast by each neighbor in the decision rule. The new Interval Valued Fuzzy k-Nearest Neighbors classifier (IVF-kNN) overcomes a drawback associated with the Fuzzy-kNN classifier, by which two particular key parameters ($kInit$ and m) are to be fixed in advance, and it provides a higher degree of flexibility throughout the whole decision process. The introduction of intervals by IVF-kNN allows us to consider different values for these parameters, obtaining as a result different degrees of membership per each training instance. This poses a novel way of introducing IVFSs in fuzzy nearest neighbor classification, substantially different to previous approaches on the use of type-2 fuzzy sets in the field [18].
- In the second stage, IVF-kNN is extended by EIVF-kNN. Besides the generalization of the $kInit$ and m parameters introduced by IVF-kNN, the evolutionary technique enables us to optimize its selection, improving in this way the accuracy of the whole model. Therefore, this extension consists of the use of evolutionary algorithms to develop an automatic method - conducted by the CHC evolutionary algorithm [19] - for optimizing the procedure to build the intervals in the IVF-kNN, adapting them to the specific data set chosen by following a *wrapper* based approach.

An experimental study is carried out to compare the performances of these and some advanced fuzzy nearest neighbor classifiers, including a previous approach on fuzzy nearest neighbor classification based on type-2 fuzzy sets [18] and an evolutionary fuzzy nearest neighbor algorithm [20]. In this study, their classification accuracy is tested over several well-known classification problems. The results are contrasted using nonparametric statistical procedures, enabling the safely confirmation of the conclusions drawn from them.

The rest of the paper is organized as follows: Section II describes the kNN and Fuzzy-kNN classifiers, highlighting the enhancements to the former introduced by the latter. Section III presents the IVF-kNN and EIVF-kNN models, as natural extensions of Fuzzy-kNN. Section IV is devoted to the experimental study performed and the analysis of its results. Finally, Section V concludes the paper.

II. K NEAREST NEIGHBORS AND FUZZY K NEAREST NEIGHBORS CLASSIFIERS

The kNN and Fuzzy-kNN classifiers require the measuring of the similarity of a new query instance (the new instance to be classified) to the instances stored in the training set. In the next step, a set of k nearest neighbors is found. Every neighbor casts a vote on the class to which the query instance should be assigned. Finally, a class is assigned to the query instance by combining these votes.

The above procedure can be formally described as follows: Let X be a training set, composed of N instances $X = \{x^0, x^1, \ldots, x^N\}$ which belong to C classes. Each instance $x^i = (x_0^i, x_1^i, \ldots, x_M^i, x_\omega^i)$ is characterized by M input attributes and one output attribute ω ($\omega \in C$). For a new query instance Q, a nearest neighbor classifier finds its k nearest neighbors in X, using a particular similarity function. Next, the class of Q is predicted as the aggregation of the class attributes ω of the k nearest neighbors.

Initially, training instances of k-NN are labeled using a *hard* scheme: The membership U of an instance x to each class of C is given by an array of values in $\{0, 1\}$, where $U_\omega(x) = 1$ and $U_c(x) = 0, c \in C, c \neq \omega$. In this scheme, each instance belongs completely to one class and does not belong to any of the rest.

In the case of Fuzzy-kNN [5], the above scheme is extended using a continuous range of membership: Memberships are quantified in $[0, 1]$, and obtained using the following membership function

$$U_c(x) = \begin{cases} 0.51 + (nn_c/kInit) * 0.49 & \text{if c = } \omega \\ (nn_c/kInit) * 0.49 & \text{otherwise} \end{cases} \tag{1}$$

where nn_c are the number of instances belonging to class c found among the $kInit$ [1] neighbors of x.

This fuzzy scheme causes instances close to the center of the classes to keep the original crisp memberships in $\{0, 1\}$, but instances close to the boundaries will spread half of their membership between the neighbors' classes.

Once a query instance Q has been presented, its k nearest neighbors are searched in the training set. Although many different similarity functions can be considered for this task, the preferred choice in nearest neighbor classification is to define it via the Euclidean distance, which should suit most classification problems if the training data is normalized in the domain $[0, 1]$. Throughout the rest of the paper we will follow this methodology: the Euclidean distance is used and the attributes are normalized.

Once the k nearest neighbors have been determined, the final output of the classifier is obtained by aggregating the votes cast by its neighbors. In the case of kNN, the votes are obtained by simply adding the memberships of the k neighbors. In the case of Fuzzy-kNN, the Euclidean norm and the memberships are weighted to produce a final vote for each class and neighbor using Equation (2):

$$V(k_j, c) = \frac{U_c(k_j) \cdot 1/(\|Q - k_j\|)^{2/(m-1)}}{\sum_{i=1}^{k} 1/(\|Q - k_i\|)^{2/(m-1)}} \tag{2}$$

where k_j is the j-th nearest neighbor ($k_j \in k$) and $m, m > 1$ is a parameter used to intensify the distances between the query Q and the training data set elements (generally $m = 2$). The votes of each neighbor are finally added to obtain the final classification, as in the case of kNN.

Thus, both classifiers obtain their final output applying the majority simple rule to the classes of the k nearest neighbors. However, in the case of Fuzzy-kNN, the use of the soft labeling scheme and the weighted votes allows it to achieve a more robust classification, particularly for instances located next to the decision boundaries, whose crisp classification would otherwise be unclear.

[1] *kInit* is usually set to an integer value between $[3, 9]$

III. IVF-kNN AND EIVF-kNN: K NEAREST NEIGHBORS CLASSIFIERS BASED ON INTERVAL VALUED FUZZY SETS AND EVOLUTIONARY ALGORITHMS

IVF-kNN is proposed as an improvement of Fuzzy-kNN through the introduction of IVFS. As a consequence, the membership values of every instance in the training set are represented as an array of intervals, depicting a more flexible representation of the typicalness of the instances in every class of the problem. Intervals are also considered in the computation of the votes cast by each of the k nearest neighbors in the decision rule. Using this approach we aim at reducing the sensitivity of the original Fuzzy-kNN classifier to the *kInit* and *m* parameters, removing the necessity of tuning their values for each specific problem.

Our second proposal, termed EIVF-kNN, introduces the use of evolutionary algorithms for optimizing the representation of both the membership values and the votes. These representations are upgraded through a second redefinition of the way in which the *kInit* and *m* parameters are interpreted. In our implementation, we have chosen the CHC evolutionary model [19] to conduct the search.

The description of our proposals is organized as follows:

- In Subsection III-A, we detail the mechanism developed to perform the computation of interval valued memberships to the classes.
- In Subsection III-B, the voting procedure of IVF-kNN and EIVF-kNN based on intervals is described.
- In Subsection III-C, we show how the votes can be combined in order to obtain the final classification.
- Finally, in Subsection III-D, the evolutionary process of EIVF-kNN is detailed, including a description of how the *kInit* and *m* parameters are adjusted, and the details necessary to implement this approach with the CHC model.

A. Computation of interval valued memberships to the classes

In Fuzzy-kNN, the definition of the memberships of the training instances is governed by Equation (1). It is designed so that the class to which an instance originally belongs obtains more than half (0.51) of the total membership, whereas the rest is shared among the rest of the classes of the problem. By searching for the $kInit$ nearest instances, local information about the relative neighborhood of the instance is considered. Therefore, this set up incorporates both expert knowledge (the ω classes already assigned in the original data) and structural knowledge, thereby obtaining a more accurate representation of the true nature of the instance than by use of the kNN classifier [5].

However, a drawback of this approach is that $kInit$ must be fixed in advance. Some rules of thumb may be considered when aiming for a proper set up, such as not setting it to an extremely low value - with $kInit = 1$ or $kInit = 2$ very few neighbors are included, and hence most of the local structural information about the data is lost - or not setting it to a very high value - which would make memberships approximately equal to the global distribution of classes in the training data, and thus discarding again the local information. Beyond this, any fixed value of $kInit$ could potentially be selected.

We argue that the use of IVFSs to represent membership of classes could provide an alternative and efficient solution to the above drawback, and therefore make the fixing of a specific value to $kInit$ superfluous. Indeed, the use of interval values for membership could accommodate the simultaneous use of different values of $kInit$. That is, Equation (1) can be parametrized with $kInit$

$$U_c(x, kInit) = \begin{cases} 0.51 + (nn_c/kInit) * 0.49 & \text{if c = } \omega \\ (nn_c/kInit) * 0.49 & \text{otherwise} \end{cases} \tag{3}$$

and then the membership of a training instance x to a class c can be represented as an interval

$$U_c(x) = [U_{c-left}(x), U_{c-right}(x)] \tag{4}$$

$$\begin{aligned} U_{c-left}(x) &= \min\left[U_c(x, kInit)\right] \\ U_{c-right}(x) &= \max\left[U_c(x, kInit)\right] \end{aligned} \tag{5}$$

with $kInit$ belonging to a particular set of values derived from the application of the considerations mentioned before. Following the same recommendations as in the case of

Fuzzy-kNN, we may consider that $kInit$ belongs to the set of integer values of the interval $[3, 9]$.

Following this scheme, a more flexible and accurate representation of the training instances is obtained:

- Instances located at the center of their respective classes, surrounded only by instances of the same class, will maintain full membership to it ($[1.0, 1.0]$) and null membership to the rest of the classes ($[0.0, 0.0]$). This is equivalent to Fuzzy-kNN and kNN.
- Instances located near the boundaries between classes, surrounded by instances of the same class, but also by some instances of other classes, will have their memberships modified as follows:
  - The lower value of the membership to ω, $U_{c-left}(x)$, may be regarded as a measure of how many neighboring instances with a class different to ω there are, and of their relevance. The higher the number of these neighboring instances to the training instance they will be, the closer to 0.51 this lower value will be.
  - The upper value of the membership to ω, $U_{c-right}(x)$, is a direct measure of how far away the first neighbor not belonging to ω is. It will be 1.0 if it is not among the first nearest neighbors (in accordance with the set up for $kInit$ chosen), and slightly lower if it is, with the specific value again dependent on the number and position of the neighboring instances not belonging to ω.
  - The lower value of the membership to the rest of classes will be 0.0, unless one of the first nearest neighbors belongs to that class. The upper value can be regarded as a relative measure of the presence of this class among the neighborhood of the training instance, never greater than 0.49.
- Instances badly identified (possibly noise), surrounded only by instances of other classes, will get only half membership to ω ($[0.51, 0.51]$) whereas the membership to the rest of the classes will be a representation of the true nature of the instances.

B. Interval valued voting procedure

The votes cast by each neighbor in the computation of the decision rule (Equation 2) can also be represented by intervals. In this expression, the parameter m can be used to vary the

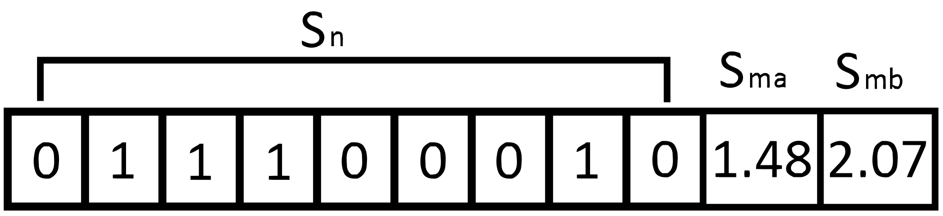


Fig. 1. A valid configuration of IVF-kNN. The values of $kInit$ chosen are 2,3,4 and 8 (from the interval $(1, 9)$) and the values of m_a and m_b are 1.48 and 2.07, respectively

influence of the neighbors, depending on the specific value chosen.

If $m = 2$, the vote of each neighbor is weighted by the reciprocal of the squared Manhattan distance. As m increases, distances between the different neighbors will be evenly weighted, and thus the relative distances will have less effect on the determination of the votes (with $m = 3$ the weight becomes the reciprocal of the Euclidean distance). Similarly, if m is decreased, the relative distances will have a greater effect, reducing the contribution of the furthest instances (as m approaches 1).

Although the choice recommended in [5] was to simply let $m = 2$, it is possible to consider this parameter in a more flexible way, by introducing intervals. This allows a range of possible values of m to be considered instead of a single one, resulting in a more general voting mechanism.

To represent this, Equation 2 becomes:

$$V(k_j, c) = U_c(k_j) \cdot D(k_j) \tag{6}$$

where

$$\begin{aligned} D(k_j) \quad = \quad & [\min(D(k_j, m_a), D(k_j, m_b)), \\ & \max(D(k_j, m_a), D(k_j, m_b))] \end{aligned} \tag{7}$$

$$D(k_j, m) = \frac{1/(\|Q - k_j\|)^{2/(m-1)}}{\sum_{i=1}^{k} 1/(\|Q - k_i\|)^{2/(m-1)}} \tag{8}$$

and m_a, m_b are the minimum and maximum values chosen for the parameter m. Note that since the elements of Equation (6) are intervals, their product must be computed as follows [21]:

$$[a_1, a_2] * [b_1, b_2] = [\min(I_j), \max(I_j)]$$
$$I_j = \{a_1 \cdot b_1, a_1 \cdot b_2, a_2 \cdot b_1, a_2 \cdot b_2\} \tag{9}$$

C. Combination of votes

After the votes have been computed, the final classification is obtained as the class with the maximum vote overall. In the case of IVF-kNN, the votes for every class are computed as

$$V(c) = \sum_{j=1}^{k} V(k_j, c) \tag{10}$$

where the addition of two intervals is obtained as follows:

$$[a_1, a_2] + [b_1, b_2] = [a_1 + b_1, a_2 + b_2] \tag{11}$$

After the votes for every class have been added, every interval is converted to a single value (the center of the interval). The final classification is obtained as the class with the highest center of interval. In the case of a tie, only the contribution of the first nearest neighbor is considered in order to discriminate between the tied classes [2].

D. Evolutionary optimization of IVF-kNN: The EIVF-kNN model

As is detailed in the description of IVF-kNN, the performance of the classifier is dependent on the selection of values considered for $kInit$ and $[m_a, m_b]$. Although proper values may be fixed experimentally, in this subsection we present an automatic method for performing an optimum selection, guided by an evolutionary algorithm.

[2]This is inspired by one of the possible tie-break procedures for kNN, where the class of the first nearest neighbor is used to break ties. Note, however, that in the case of Fuzzy-kNN and IVF-kNN such ties are very unlikely.

For parameter m_a and m_b, the optimization procedure needs to find two real values within a reasonable range (in accordance with the definition of Equation 8). However, the selection of a set of values for $kInit$ may be more sophisticated.

Instead of just fixing this selection to an interval of integer values (for example, $\{3, 4, 5, 6, 7, 8, 9\}$), as it is recommended for the basic IVF-kNN model, it is possible to define an optimization procedure for choosing specific, non-correlative values.

Figure 1 shows an example of a valid configuration for IVF-kNN. It is is composed of a binary array of S_n digits, in which a value $S_n = 1$ shows that n is chosen as a value for $kInit$, while a value $S_n = 0$ shows that n is not chosen, and two real values, corresponding to m_a and m_b.

In the example illustrated in Figure 1, the interval [1.48,2.07] has been chosen for the m parameter. The values 2,3,4 and 8 have been chosen for the $kInit$ parameter. Note that this set-up is similar to the configuration of IVF-kNN with $kInit$ in (2,8), but it will not produce the same memberships, as the values 5,6 and 7 are not considered. Thus, this configuration could be useful in such cases where the memberships should be constrained to the 4 nearest neighbors of each training instance, but then extended by considering an instance at a greater distance (the eighth neighbor).

Nevertheless, with this representation, an optimization algorithm may be applied in order to obtain the best possible configuration for the specific data analyzed. In our case, we recommend the use of the CHC evolutionary algorithm [19], a robust generational genetic algorithm which involves the combination of an advanced selection strategy with a very highly selective pressure, and several components inducing a strong diversity.

The CHC algorithm is configured as follows:

- Representation of solutions: Binary chromosomes of S_n bits (representing the values chosen for $kInit$) and 2 real-coded values (representing the m_a and m_b values). Figure 1 is, in fact, a valid chromosome of the algorithm.
- Initialization: All chromosomes are initialized randomly. Real values are initialized into the interval $]1, 4]$ (m cannot take a value of 1 due to the definition of Equation (8), and should never be set to a very high value).
- Crossover operator: HUX and BLX-0.5. The HUX operator (the classic crossover op-

erator of CHC, which interchanges half of the non-matching genes of each parent) is used for the binary part of the solutions, whereas the BLX-0.5 operator [22] is used for crossing the real coded values.

- Fitness function: IVF-kNN is used as a *wrapper*. To evaluate a solution, IVF-kNN is configured with the parameters represented by the solution, and its accuracy over the training data (using a *leave-one-out* validation scheme) is measured. This accuracy is considered to be the fitness of the solution.

The rest of the elements follow the same configuration as in the original definition of CHC.

In summary, the application of CHC to optimize the configuration of IVF-kNN enables us to obtain a suitable configuration, adapted to the training data available, without the necessity of a specific set-up provided by the practitioner. This increases the adaptability of the whole model, EIVF-kNN, for general supervised learning problems, providing that the training data available is representative of the problem considered.

IV. Experimental study

An experimental study has been carried out to test the performance of EIVF-kNN and its basic version, IVF-kNN. The experiments will involve several well-known classification problems and various state of the art algorithms in fuzzy nearest neighbor classification. Section IV-A describes the experimental framework in which all the experiments have been carried out. Section IV-B shows the results achieved by IVF-kNN. Finally, Section IV-C shows the results achieved by EIVF-kNN.

A. Experimental framework

The experiments has been conducted over 44 classification data sets, whose partitions are available at the KEEL-Dataset repository[3] [23], [24]. Table I summarizes the following characteristics: number of instances (**#Ins.**), number of attributes **(#At.)** and number of classes **(#Cl.)**. None of the data sets includes missing values, and no nominal (discrete) attributes have been considered. All Attribute values have been normalized at $[0, 1]$ and a 10-folds cross validation procedure has been followed throughout the experiments.

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

TABLE I
DATA SETS CONSIDERED IN THE STUDY

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

Besides EIVF-kNN and IVF-kNN, we have considered 7 comparison algorithms, which can be categorized as follows:

- **Classical algorithms**: Three classical algorithms have been chosen: kNN, Fuzzy-kNN and IT2FKNN. IT2FKNN [18] is a fuzzy nearest neighbor classifier based on the use of interval type-2 fuzzy sets to represent the membership to the classes of the training instances. In this way, several values for the parameter $kInit$ are considered, thus including flexibility in the representation of the memberships computed by considering distinct choices. Type-2 fuzzy sets are then built considering all the different memberships computed, and then a type reduction operation is performed to obtain a final, combined value, representative of all the choices considered initially. The algorithm is similar to Fuzzy-kNN in respect to the rest of the phases.

- **Recent proposals**: Three recent proposals have been selected as a representation of the current state of the art: CFKNN [25] is a preprocessing method for Fuzzy-KNN inspired by the classic Hart's condensing rule - the CNN algorithm [26]. Before the application of Fuzzy-kNN to classify new instances, instances of the training set are removed depending on an entropy measure which determines whether they are removed or kept. FRNN-FRS [27] is a fuzzy nearest neighbor algorithm that makes use of rough uncertainties, and performs a classification based on the best value obtained for the lower and upper approximations of every nearest neighbor, choosing the class of the nearest neighbor with the best value (not necessarily the nearest). Finally, FRNN-VQRS [27] is a method similar to FRNN-FRS, but uses vaguely quantified rough sets to model the lower and upper approximations of the nearest neighbors, instead of fuzzy rough sets.
- **Evolutionary approaches**: We have selected GAFuzzy-kNN [20] as a proper evolutionary approach. GAFuzzy-kNN is a wrapper-based Fuzzy-kNN classifier. It works by estimating the best possible values for the parameters m and $kInit$ by means of a binary genetic algorithm. The search is conducted by a fitness function, which computes the accuracy of an underlying Fuzzy-KNN classifier as the fitness value.

Table II shows the parameters configuration selected for each algorithm. The parameter k is chosen for each algorithm in the {3,5,7,9} range [4], selecting the best performing value. That is, every algorithm has been tested considering each different value of k, and the best value found (the one that maximizes the average accuracy) has been chosen.

For IVF-kNN, the initial intervals for $kInit$ and m have been determined according to preliminary experiments. For EIVF-kNN, the range of values of $kInit$ allowed and the range in which the maximum and minimum value of m can be established have been set-up according to the range suggested in [20], in which GAFuzzy-kNN was presented. The population size and the number of evaluations have been also chosen as in [20], for the sake of a fair comparison. The rest of the parameters have been set up following the recommendations given by the authors of each technique. Euclidean distance has been used as the similarity

[4]Note that k=1 has been excluded since most of the fuzzy nearest neighbor algorithms would become the 1NN rule. Also, no higher values of k have been chosen because most of the classifiers would degenerate to a majority classifier, discarding the locality capabilities of nearest neighbor algorithms

TABLE II
PARAMETERS CONFIGURATION OF THE ALGORITHMS

Algorithm	Parameters
IVF-kNN	k value: 7, $kInit$: (3,9), m: [1.5,2]
kNN	k value: 7
Fuzzy-kNN	k value: 5, $kInit$: 3, m: 2
IT2FKNN	k value: 7, $kInit$: {1,3,5,7,9}, m: 2
CFKNN	k value: 3, α: 0.6
FRNN-FRS	k value: 3
FRNN-VQRS	k value: 5
EIVF-kNN	k value: 9, $kInit$ range: (1,32), m range: [1,4], S_n:32, Population size: 50, Evaluations: 500
GAFuzzy-kNN	k value: 5, $kInit$: 3, m: 2, Population size: 50, Evaluations: 500, Crossover probability: 0.8, Mutation probability: 0.01

measure in all the experiments.

The results obtained in all the experiments will be contrasted through the use of nonparametric statistical tests [28], [29]. Specifically, we will use the Wilcoxon signed-ranks test [30] for pairwise comparisons, and the Friedman test [31] (together with the Holm procedure [32] as post-hoc) for performing multiple comparisons. More information about these statistical procedures specifically designed for use in the field of Machine Learning can be found at the SCI2S thematic public website on *Statistical Inference in Computational Intelligence and Data Mining* [5].

B. Analysis of the IVF-kNN algorithm

In this first study, we have included IVF-kNN, the classical algorithms kNN, Fuzzy-kNN and IT2FKNN, and the recent approaches CFKNN, FRNN-FRS, FRNN-VQRS. Table III shows the accuracy results obtained in the test classification phase. For each data set, the best results obtained have been highlighted in **bold**. The table also includes the average accuracy obtained over the 44 datasets.

Considering these results, we can make the following analysis:

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

TABLE III
RESULTS OF THE FIRST STUDY: ANALYSIS OF THE IVF-kNN ALGORITHM

Data sets	IVF-kNN	kNN	Fuzzy-kNN	IT2FKNN	CFKNN	FRNN-FRS	FRNN-VQRS
Appendicitis	87.00	**87.91**	**87.91**	87.91	82.27	83.09	83.09
Balance	88.64	88.48	88.63	**88.80**	70.72	77.76	75.52
Banana	89.30	**89.58**	89.19	89.51	56.36	87.45	87.43
Bands	70.08	69.75	**71.31**	69.51	59.55	67.36	65.77
Bupa	64.05	62.53	62.50	**65.21**	56.90	60.65	60.33
Cleveland	**57.31**	56.92	55.97	56.97	53.30	55.34	55.35
Dermatology	**96.34**	**96.34**	96.33	**96.34**	95.79	94.65	93.53
Ecoli	**82.76**	82.45	82.46	82.45	72.33	80.69	80.69
Glass	**72.61**	66.83	72.57	72.11	45.42	70.86	71.33
Haberman	68.61	**69.90**	67.34	68.96	65.06	64.04	64.04
Hayes-roth	65.63	28.75	65.63	63.75	47.50	**75.00**	73.13
Heart	80.00	79.26	80.37	78.52	**81.48**	77.04	73.33
Hepatitis	84.67	**89.19**	83.42	84.67	83.51	40.89	25.32
Ionosphere	**84.32**	84.03	84.04	**84.32**	72.08	82.03	81.17
Iris	94.67	**96.00**	**96.00**	95.33	94.00	95.33	**96.00**
Led7Digit	71.40	43.40	71.60	71.40	**73.20**	61.80	61.00
Mammographic	79.28	**81.71**	80.37	79.54	79.02	74.47	74.46
Marketing	**30.92**	29.51	30.79	30.81	28.97	27.68	27.41
Monk-2	84.46	89.16	**89.69**	82.50	82.27	77.69	76.32
Movement	82.22	72.50	82.22	81.11	55.56	**86.39**	**86.39**
New Thyroid	93.98	92.58	93.98	93.98	92.08	**97.23**	**97.23**
Page-blocks	**96.05**	95.47	95.91	95.87	76.88	95.69	95.60
Penbased	99.17	99.13	99.24	99.14	82.16	**99.30**	99.29
Phoneme	**90.01**	87.75	89.36	89.65	73.83	**90.01**	89.91
Pima	73.32	72.93	72.93	**73.58**	71.11	71.11	71.24
Ring	59.86	67.46	60.77	58.84	71.58	78.85	**79.45**
Satimage	90.29	90.52	**90.55**	90.12	78.52	89.90	89.62
Segment	96.28	94.81	96.36	96.15	84.46	**96.49**	96.36
Sonar	83.10	80.21	81.64	82.14	68.83	**86.98**	86.02
Spambase	**91.28**	89.34	91.15	90.93	83.58	89.49	89.06
Spectfheart	76.05	**77.58**	74.23	77.55	56.58	68.55	67.42
Tae	**67.00**	45.08	66.29	**67.00**	55.08	63.04	63.04
Texture	98.40	98.31	98.53	98.35	77.95	**99.11**	99.05
Thyroid	93.97	**93.99**	93.97	93.94	43.93	90.89	90.64
Titanic	78.06	76.24	75.69	78.15	75.06	78.92	**79.06**
Twonorm	97.01	**97.07**	97.01	97.04	96.82	94.54	94.24
Vehicle	71.64	**72.34**	70.81	71.16	42.92	68.79	68.79
Vowel	97.78	88.69	97.47	97.07	39.29	**99.39**	99.29
Wdbc	**97.36**	97.18	96.65	97.18	94.19	94.89	94.54
Wine	96.60	96.63	96.01	**97.19**	96.05	94.41	93.86
Winequality-red	**69.04**	55.29	67.98	68.73	33.83	65.10	64.79
Winequality-white	**68.60**	50.92	67.52	68.38	32.11	65.66	65.40
Wisconsin	96.96	**97.25**	**97.25**	96.96	96.22	94.76	94.46
Yeast	59.71	57.49	58.89	**59.91**	51.08	51.62	51.76
Average	**81.27**	78.15	81.10	81.11	68.85	78.75	77.99

- IVF-kNN effectively improves the accuracy with respect to the kNN classifier, improving its performance by more than a 3% on average.
- IVF-kNN achieves a better average result than all the classical techniques. Neither Fuzzy-kNN nor IT2FKNN are able to obtain a higher accuracy, on average.
- When compared with the recent algorithms, we can also highlight IVF-kNN as a very accurate classifier, which obtains a better average result than CFKNN, FRNN-FRS and FRNN-VQRS.

TABLE IV
RESULTS OF THE WILCOXON TEST - CLASSICAL APPROACHES

Comparison	R^+	R^-	p-value
IVF-kNN vs Fuzzy-kNN	636.5	309.5	0.04833
IVF-kNN vs IT2FKNN	616.0	330.0	0.08536
IVF-kNN vs kNN	674.0	272.0	0.01441

TABLE V
RESULTS OF THE FRIEDMAN & HOLM TESTS - RECENT APPROACHES

Algorithm	Rank	Holm p-value
IVF-kNN	1.5795	-
CFKNN	3.4205	0.00000
FRNN-FRS	2.2386	0.01664
FRNN-VQRS	2.7614	0.00002
Friedman p-value	$\leq 10^{-8}$	

These results have been contrasted by using nonparametric statistical procedures. For contrasting the comparison with classical algorithms, we have chosen the Wilcoxon test as a pairwise procedure. Table IV shows the results of the test, including the ranks obtained in each comparison (R^+ and R^-) and the p-value associated.

The results of this test enables us to confirm that IVF-kNN is significantly better than the basic kNN, thus becoming a proper enhancement of this classic algorithm. It also overcomes statistically the Fuzzy-kNN, which reveals that the way in which IVF-kNN manages the fuzzy memberships and the votes is more appropriate. Finally, it is possible to highlight that the differences between IT2FKNN and IVF-kNN are also significant, which means that the latter performs better when determining the fuzzy memberships than using interval type-2 fuzzy sets.

A second statistical analysis can be performed to contrast the comparison between IVF-kNN and the recent approaches to fuzzy nearest neighbor classification. In this case, the Friedman test provides us with a procedure for simultaneously contrasting the comparisons of IVF-kNN with all the recent approaches.

Table V shows the results obtained by the application of the Friedman test and the Holm post-hoc procedure. The Friedman p-value is lower than 10^{-8}, which means that significant differences are found among IVF-kNN, CFKNN, FRNN-FRS and FRNN-VQRS. The Holm procedure reports a very low p-value in all the pairwise comparisons of the control algorithm, IVF-kNN, with the rest of techniques, which indicates that it significantly outperforms all these recent approaches.

In summary, these results show us that IVF-kNN is a very competitive fuzzy nearest neighbor classifier. The use of interval values for defining memberships and for computing the neighbors' votes produces better results than those using Fuzzy-kNN in its original definition. Furthermore, IVF-kNN also achieves an accuracy improvement over the IT2FKNN classifier, a type-2 fuzzy nearest neighbor classifier. The comparison has been completed including three recent approaches, CFKNN, FRNN-FRS and FRNN-VQRS, whose results are also improved by IVF-kNN.

C. Analysis of the EIVF-kNN algorithm

A second study has been performed, aimed at analyzing whether the improvements proposed in the EIVF-kNN model are effective for enhancing the performance of IVF-kNN. Moreover, GAFuzzy-kNN is also included here as a strong evolutionary fuzzy nearest neighbor classifier, to complete the comparison.

Table VI shows the accuracy results obtained in this second study, highlighting the best results obtained in each data set in **bold** and including the average accuracy obtained over the 44 datasets. These results have also been contrasted with the Wilcoxon test, the results of which are shown in Table VII.

The results shown in Tables VI and VII shows that EIVF-kNN improves the results obtained by IVF-kNN, thus justifying the usefulness of the evolutionary method for determining which values of the $kInit$ and m parameters to consider during the construction of the classifier. EIVF-kNN also shows a better performance than GAFuzzy-kNN, a representative approach to evolutionary fuzzy nearest neighbor classification. These results are corroborated by the Wilcoxon test, contrasting these conclusions.

V. Conclusion

In this paper we have proposed a new evolutionary interval valued nearest neighbor classifier. IVFS are chosen as an appropriate tool for representing the instances' memberships to the

TABLE VI
RESULTS OF THE SECOND STUDY: ANALYSIS OF THE EIVF-kNN ALGORITHM

Data sets	IVF-kNN	EIVF-kNN	GAFuzzy-kNN
Appendicitis	**87.00**	85.18	86.09
Balance	88.64	**89.12**	88.00
Banana	89.30	**89.85**	89.28
Bands	70.08	**70.34**	69.63
Bupa	**64.05**	63.99	61.68
Cleveland	**57.31**	56.91	53.55
Dermatology	96.34	95.78	**96.35**
Ecoli	82.76	**83.04**	82.48
Glass	72.61	71.98	**73.38**
Haberman	68.61	**71.58**	68.95
Hayes-roth	**65.63**	65.00	64.38
Heart	80.00	**81.11**	78.52
Hepatitis	**84.67**	82.90	83.42
Ionosphere	84.32	84.89	**87.75**
Iris	94.67	94.00	**95.33**
Led7Digit	71.40	**71.80**	71.20
Mammographic	79.28	**79.89**	79.53
Marketing	30.92	**30.98**	29.51
Monk-2	84.46	83.45	**92.42**
Movement	82.22	84.44	**84.72**
New Thyroid	93.98	**96.34**	95.89
Page-blocks	96.05	**96.20**	96.05
Penbased	99.17	**99.34**	99.32
Phoneme	90.01	**90.30**	89.60
Pima	73.32	73.57	**74.36**
Ring	59.86	**71.41**	63.39
Satimage	90.29	**90.71**	90.40
Segment	96.28	**96.58**	96.54
Sonar	83.10	83.07	**84.55**
Spambase	91.28	**91.45**	91.26
Spectfheart	76.05	**77.54**	75.00
Tae	67.00	**70.96**	66.96
Texture	98.40	**99.04**	98.91
Thyroid	93.97	**94.00**	93.93
Titanic	78.06	**78.69**	78.51
Twonorm	97.01	**97.08**	96.97
Vehicle	**71.64**	70.81	70.33
Vowel	97.78	**99.19**	98.79
Wdbc	**97.36**	97.01	96.48
Wine	96.60	**97.16**	95.49
Winequality-red	**69.04**	68.48	67.10
Winequality-white	**68.60**	67.99	66.40
Wisconsin	**96.96**	96.81	**96.96**
Yeast	59.71	**59.91**	57.89
Average	81.27	**81.81**	81.30

TABLE VII
RESULTS OF THE WILCOXON TEST - EVOLUTIONARY APPROACHES

Comparison	R^+	R^-	p-value
EIVF-kNN vs IVF-kNN	639.0	351.0	0.09422
EIVF-kNN vs GAFuzzy-kNN	713.5	276.5	0.00998

different classes of the problem. They also enable our classifier to represent several votes as a single interval, thus giving more flexibility to the decision rule computation, and ultimately, improving the generalization capabilities of the nearest neighbor rule. The CHC evolutionary algorithm has been incorporated to the model, in order to ease the task of properly searching for the best intervals of the $kInit$ and m parameters, and thus further improving its accuracy. This has also been corroborated experimentally and it may be concluded that IVF-kNN and EIVF-kNN are significantly more accurate than a selection of classical and new approaches in the state of the art of fuzzy nearest neighbor classification.

ACKNOWLEDGMENT

Supported by the Projects TIN2011-28488, P10-TIC-06858 and P11-TIC-9704. J. Derrac holds an FPU scholarship from the Spanish Ministry of Education, and was a Research Visitor in DIGITS (DMU, Leicester) in 2012.

REFERENCES

[1] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[2] X. Wu and V. Kumar, Eds., *The Top Ten Algorithms in Data Mining*, ser. Data Mining and Knowledge Discovery. Chapman & Hall/CRC, 2009.

[3] A. N. Papadopoulos and Y. Manolopoulos, *Nearest Neighbor Search: A Database Perspective*. Springer-Verlag Telos, 2004.

[4] G. Shakhnarovich, T. Darrell, and P. Indyk, Eds., *Nearest-Neighbor Methods in Learning and Vision: Theory and Practice*. The MIT Press, 2006.

[5] J. M. Keller, M. R. Gray, and J. A. Givens, "A fuzzy k-nearest neighbor algorithm," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 15, no. 4, pp. 580–585, 1985.

[6] D. Cabello, S. Barro, J. Salceda, R. Ruiz, and J. Mira, "Fuzzy k-nearest neighbor classifiers for ventricular arrhytmia detection," *International Journal of Biomedical Computing*, vol. 27, pp. 77–93, 1991.

[7] M. A. Chikh, M. Saidi, and N. Settouti, "Diagnosis of diabetes diseases using an artificial immune recognition system2 (AIRS2) with fuzzy k-nearest neighbor," *Journal of medical systems*, vol. 36, pp. 2721–2729, 2012.

[8] H.-L. Chen, B. Yang, G. Wang, J. Liu, X. Xu, S. Wang, and D. Liu, "A novel bankruptcy prediction model based on an adaptive fuzzy k-nearest neighbor method," *Knowledge Based Systems*, vol. 24, no. 8, pp. 1348–1359, 2011.

[9] Y. Huang and Y. Li, "Prediction of protein subcellular locations using fuzzy k-nn method," *Bioinformatics*, vol. 20, no. 1, pp. 21–28, 2004.

[10] T. Liao and D. Li, "Two manufacturing applications of the fuzzy k-nn algorithm," *Fuzzy Sets and Systems*, vol. 92, pp. 289–303, 1997.

[11] H. B. Sola, "Indicator of inclusion grade for interval-valued fuzzy sets. application to approximate reasoning based on interval-valued fuzzy sets," *International Journal of Approximated Reasoning*, vol. 23, no. 3, pp. 137–209, 2000.

[12] J. M. Mendel, "Advances in type-2 fuzzy sets and systems," *Information Sciences*, vol. 117, pp. 84–110, 2007.

[13] D. Dubois and H. Prade, "An introduction to bipolar representations of information and preference," *International Journal of Intelligent Systems*, vol. 23, no. 8, pp. 866–877, 2008.

[14] H. Bustince, M. Pagola, E. Barrenechea, P. Melo-Pinto, P. Couto, H. Tizhoosh, and J. Montero, "Ignorance functions. an application to the calculation of the threshold in prostate ultrasound images," *Fuzzy Sets and Systems*, vol. 161, no. 1, pp. 20–36, 2010.

[15] J. A. Sanz, A. Fernández, H. Bustince, and F. Herrera, "Improving the performance of fuzzy rule-based classification systems with interval-valued fuzzy sets and genetic amplitude tuning," *Information Sciences*, vol. 180, no. 19, pp. 3674–3685, 2010.

[16] ——, "A genetic tuning to improve the performance of fuzzy rule-based classification systems with interval-valued fuzzy sets: Degree of ignorance and lateral position," *International Journal of Approximate Reasoning*, vol. 52, no. 6, pp. 751–766, 2011.

[17] ——, "IVTURS: a linguistic fuzzy rule-based classification system based on a new Interval-Valued fuzzy reasoning method with TUning and Rule Selection," *IEEE Transactions on Fuzzy Systems, In press*, 2013.

[18] F. Chung-Hoon and C. Hwang, "An interval type-2 fuzzy k-nearest neighbor," in *Proceedings of the 12th IEEE International Conference on Fuzzy Systems (IEEE FUZZY'03), St. Louis, Missouri, USA, May 25-28*, 2003, pp. 802–807.

[19] L. J. Eshelman, "The CHC adaptative search algorithm: How to have safe search when engaging in nontraditional genetic recombination," in *Foundations of Genetic Algorithms*, G. J. E. Rawlins, Ed. San Mateo, California.: Morgan Kaufmann, 1991, pp. 265–283.

[20] X. Hu and C. Xie, "Improving fuzzy k-nn by using genetic algorithm," *Journal of Computational Information Systems*, vol. 1, no. 2, pp. 203–213, 2005.

[21] M. Hanss, *Applied Fuzzy Arithmetic. An Introduction with Engineering Applications*. Springer-Verlag, 2005.

[22] L. J. Eshelman and J. D. Schaffer, "Real-coded genetic algorithms and interval-schemata," in *Proceedings of the Second Workshop on Foundations of Genetic Algorithms. Vail, Colorado, USA, July 26-29*, 1992, pp. 187–202.

[23] J. Alcalá-Fdez, L. Sánchez, S. García, M. J. del Jesus, S. Ventura, J. M. Garrell, J. Otero, C. Romero, J. Bacardit, V. M. Rivas, J. C. Fernández, and F. Herrera, "KEEL: a software tool to assess evolutionary algorithms for data mining problems," *Soft Computing*, vol. 13, no. 3, pp. 307–318, 2008.

[24] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, and F. Herrera, "Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework," *Journal of Multiple-Valued Logic and Soft Computing*, vol. 17, no. 2-3, 2011.

[25] J.-H. Zhai, N. Li, and M.-Y. Zhai, "The condensed fuzzy k-nearest neighbor rule based on sample fuzzy entropy," in *Proceedings of the 2011 International Conference on Machine Learning and Cybernetics (ICMLC'11), Guilin, China, July 10-13*, 2011, pp. 282–286.

[26] P. E. Hart, "The condensed nearest neighbor rule," *IEEE Transactions on Information Theory*, vol. 18, pp. 515–516, 1968.

[27] R. Jensen and C. Cornelis, “Fuzzy-rough nearest neighbour classification,” *Transactions on Rough Sets*, vol. 13, pp. 56–72, 2011.

[28] S. García and F. Herrera, “An extension on Statistical Comparisons of Classifiers over Multiple Data Sets for all pairwise comparisons,” *Journal of Machine Learning Research*, vol. 9, pp. 2677–2694, 2008.

[29] S. García, A. Fernández, J. Luengo, and F. Herrera, “Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power,” *Information Sciences*, vol. 180, pp. 2044–2064, 2010.

[30] F. Wilcoxon, “Individual comparisons by ranking methods,” *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945.

[31] M. Friedman, “The use of ranks to avoid the assumption of normality implicit in the analysis of variance,” *Journal of the American Statistical Association*, vol. 32, pp. 674–701, 1937.

[32] S. Holm, “A simple sequentially rejective multiple test procedure,” *Scandinavian Journal of Statistics*, vol. 6, pp. 65–70, 1979.

Bibliography

[Aha97] Aha D. W. (Ed.) (1997) *Lazy Learning.* Springer.

[AK09] Ahn H. and Kim K. (2009) Bankruptcy prediction modeling with hybrid case-based reasoning and genetic algorithms approach. *Applied Soft Computing* 9: 599–607.

[AKA91] Aha D. W., Kibler D., and Albert M. K. (1991) Instance-based learning algorithms. *Machine Learning* 6: 37–66.

[Alb05] Alba E. (2005) *Parallel Metaheuristics: A New Class of Algorithms.* Wiley-Interscience.

[Alp10] Alpaydin E. (2010) *Introduction to Machine Learning.* The MIT Press, 2 edition.

[AMS97] Atkeson C. G., Moore A. W., and Schaal S. (1997) Locally weighted learning. *Artificial Intelligence Review* 11: 11–73.

[Ata86] Atanassov K. (1986) Intuitionistic fuzzy sets. *Fuzzy Sets and Systems* 20: 87–96.

[BBBE11] Bizer C., Boncz P. A., Brodie M. L., and Erling O. (2011) The meaningful use of big data: four perspectives - four challenges. *SIGMOD Record* 40(4): 56–60.

[BHS09] Bell G., Hey T., and Szalay A. (2009) Beyond the data deluge. *Science* 323: 1297–1298.

[BPM04] Batista G. E. A. P. A., Prati R. C., and Monard M. C. (2004) A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explorations* 6(1): 20–29.

[CCK07] Cock M. D., Cornelis C., and Kerre E. (2007) Fuzzy rough sets: The forgotten step. *IEEE Transactions on Fuzzy Systems* 15(1): 121–130.

[CGH08] Cano J. R., García S., and Herrera F. (2008) Subgroup discover in large size data sets preprocessed using stratified instance selection for increasing the presence of minority classes. *Pattern Recognition Letters* 29(16): 2156–2164.

[CGI09] Cervantes A., Galván I. M., and Isasi P. (2009) Ampso: A new particle swarm method for nearest neighborhood classification. *IEEE Transactions on Systems, Man, and Cybernetics–Part B: Cybernetics* 39(5): 1082–1091.

[CH67] Cover T. M. and Hart P. E. (1967) Nearest neighbor pattern classification. *IEEE Transactions on Information Theory* 13(1): 21–27.

[CHL03] Cano J. R., Herrera F., and Lozano M. (2003) Using evolutionary algorithms as instance selection for data reduction in KDD: An experimental study. *IEEE Transactions on Evolutionary Computation* 7(6): 561–575.

[CHL05] Cano J. R., Herrera F., and Lozano M. (2005) Stratification for scaling up evolutionary prototype selection. *Pattern Recognition Letters* 26(7): 953–963.

[CHL07] Cano J. R., Herrera F., and Lozano M. (2007) Evolutionary stratified training set selection for extracting classification rules with trade off precision-interpretability. *Data & Knowledge Engineering* 60(1): 90–108.

[CHLG08] Cano J. R., Herrera F., Lozano M., and García S. (2008) Making cn2-sd subgroup discovery algorithm scalable to large size data sets using instance selection. *Expert Systems with Applications* 35(4): 1949–1965.

[CJHS10] Cornelis C., Jensen R., Hurtado G., and Slezak D. (2010) Attribute selection with fuzzy decision reducts. *Information Sciences* 180: 209–224.

[Das98] Dasgupta D. (1998) *Artificial immune systems and their applications.* Springer Verlag.

[DCK07] De Cock M., Cornelis C., and Kerre E. E. (2007) Fuzzy rough sets: The forgotten step. *IEEE Transactions on Fuzzy Systems* 15(1): 121–130.

[DGH10] Derrac J., García S., and Herrera F. (2010) Stratified prototype selection based on a steady-state memetic algorithm: A study of scalability. *Memetic Computing* 2(3): 183–199.

[DGW02] Domingo C., Gavaldà R., and Watanabe O. (2002) Adaptive sampling methods for scaling up knowledge discovery algorithms. *Data Mining & Knowledge Discovery* 6(2): 131–152.

[DHS00] Duda R., Hart P., and Stork D. (2000) *Pattern Classification.* Wiley-Interscience, John Wiley & Sons, Southern Gate, Chichester, West Sussex, England, 2 edition.

[DP97] Domingos P. and Pazzani M. J. (1997) On the optimality of the simple bayesian classifier under zero-one loss. *Machine Learning* 29(2-3): 103–130.

[ES03] Eiben A. E. and Smith J. E. (2003) *Introduction to Evolutionary Computing.* Natural Computing. Springer-Verlag.

[FH51] Fix E. and Hodges J. (February 1951) Discriminatory analysis, nonparametric discrimination: Consistency properties. Technical Report 4, USAF School of Aviation Medicine.

[FPSS96] Fayyad U. M., Piatetsky-Shapiro G., and Smyth P. (1996) From data mining to knowledge discovery in databases. *AI Magazine* 17(3): 37–54.

[Fre02] Freitas A. A. (2002) *Data Mining and Knowledge Discovery with Evolutionary Algorithms.* Springer-Verlag.

[Gar08] Garain U. (2008) Prototype reduction using an artificial immune model. *Pattern Analysis & Applications* 11(3-4): 353–363.

[GCH08] García S., Cano J. R., and Herrera F. (2008) A memetic algorithm for evolutionary prototype selection: A scaling up approach. *Pattern Recognition* 41(8): 2693–2709.

[GDCH12] García S., Derrac J., Cano J. R., and Herrera F. (2012) Prototype selection for nearest neighbor classification: Taxonomy and empirical study. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34(3): 417–435.

[GE03] Guyon I. and Elisseeff A. (2003) An introduction to variable and feature selection. *Journal of Machine Learning Research* 3: 1157–1182.

[GGNZ06] Guyon I., Gunn S., Nikravesh M., and Zadeh L. A. (Eds.) (2006) *Feature Extraction: Foundations and Applications*. Springer.

[GH09] García S. and Herrera F. (2009) Evolutionary undersampling for classification with imbalanced datasets: Proposals and taxonomy. *Evolutionary Computation* 17(3): 275–306.

[GJ05] Ghosh A. and Jain L. C. (Eds.) (2005) *Evolutionary Computation in Data Mining*. Springer-Verlag.

[GLS$^+$12] García S., Luengo J., Sáez J. A., López V., and Herrera F. (2012) A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning. *IEEE Transactions on Knowledge and Data Engineering, In press* .

[GP07] Gagné C. and Parizeau M. (2007) Co-evolution of nearest neighbor classifiers. *International Journal of Pattern Recognition and Artificial Intelligence* 21(5): 921–946.

[GPdCOB10] García-Pedrajas N., del Castillo J. A. R., and Ortiz-Boyer D. (2010) A cooperative coevolutionary algorithm for instance selection for instance-based learning. *Machine Learning* 78(3): 381–420.

[GPPR12] García-Pedrajas N. and Pérez-Rodríguez J. (2012) Multi-selection of instances: A straightforward way to improve evolutionary instance selection. *Applied Soft Computing* 12(11): 3590–3602.

[Han05] Han J. (2005) *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[HG09] He H. and Garcia E. A. (2009) Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering* 21: 1263–1284.

[HLL02] Ho S.-Y., Liu C.-C., and Liu S. (2002) Design of an optimal nearest neighbor classifier using an intelligent genetic algorithm. *Pattern Recognition Letters* 23(13): 1495–1503.

[HTF09] Hastie T., Tibshirani R., and Friedman J. (2009) *The Elements of Statistical Learning: Data Mining, Inference and Prediction*. Springer, New York Berlin Heidelberg, 2 edition.

[INN01] Ishibuchi H., Nakashima T., and Nii M. (2001) Learning of neural networks with ga-based instance selection. In *Proceedings of the 20th North American Fuzzy Information Processing Society International Conference (NAFIPS'01), Vancouver, Canada, July 25-28*, pp. 2102–2107.

[JC09] Jensen R. and Cornelis C. (2009) Fuzzy-rough instance selection. In *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2010), Barcelona, Spain, July 18-23*, pp. 1776–1782.

[Jol02] Jolliffe I. T. (2002) *Principal Component Analysis.* Springer, 2 edition.

[Jow83] Jowik A. (1983) A learning scheme for a fuzzy k-nn rule. *Pattern Recognition Letters* 1: 287–289.

[KB98] Kuncheva L. I. and Bedzek J. C. (1998) Nearest prototype classification: Clustering, genetic algorithms, or random search? *IEEE Transactions on Systems, Man, and Cybernetics* 28(1): 160–164.

[KES01] Kennedy J., Eberhart R. C., and Shi Y. (2001) *Swarm intelligence.* Morgan Kaufmann Publishers.

[KGG85] Keller J. M., Gray M. R., and Givens J. A. (1985) A fuzzy k-nearest neighbor algorithm. *IEEE Transactions on Systems, Man, and Cybernetics* 15(4): 580–585.

[Kim06] Kim K. (2006) Artificial neural networks with evolutionary instance selection for financial forecasting. *Expert Systems with Applications* 30(3): 519–526.

[KJ97] Kohavi R. and John G. H. (1997) Wrappers for feature subset selection. *Artificial Intelligence* 97(1-2): 273–324.

[Kun95] Kuncheva L. I. (1995) Editing for the k–nearest neighbors rule by a genetic algorithm. *Pattern Recognition Letters* 16: 809–814.

[LFMTH12] López V., Fernandez A., Moreno-Torres J. G., and Herrera F. (2012) Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. open problems on intrinsic data characteristics. *Expert Systems with Applications* 39(7): 6585–6608.

[LM01] Liu H. and Motoda H. (Eds.) (2001) *Instance Selection and Construction for Data Mining.* The Springer International Series in Engineering and Computer Science. Springer.

[LM07] Liu H. and Motoda H. (Eds.) (2007) *Computational Methods of Feature Selection.* Chapman & Hall/Crc Data Mining and Knowledge Discovery Series. Chapman & Hall/Crc.

[LY05] Liu H. and Yu L. (2005) Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on Knowledge and Data Engineering* 17(3): 1–12.

[MD01] Mjolsness E. and DeCoste D. (2001) Machine learning for science: State of the art and future prospects. *Science* 293: 2051–2055.

[Men07] Mendel J. M. (2007) Advances in type-2 fuzzy sets and systems. *Information Sciences* 117: 84–110.

[NL09] Nanni L. and Lumini A. (2009) Particle swarm optimization for prototype reduction. *Neurocomputing* 72(4-6): 1092–1097.

[Paw82] Pawlak Z. (1982) Rough sets. *International Journal of Computer and Information Sciences* 11(5): 341–356.

[PF09] Pappa G. L. and Freitas A. A. (2009) *Automating the Design of Data Mining Algorithms: An Evolutionary Computation Approach.* Natural computing. Springer.

[PJ00] Potter M. A. and Jong K. A. D. (2000) Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary Computation* 8(1): 1–29.

[PK99] Provost F. J. and Kolluri V. (1999) A survey of methods for scaling up inductive algorithms. *Data Mining & Knowledge Discovery* 3(2): 131–169.

[Pyl99] Pyle D. (1999) *Data Preparation for Data Mining.* The Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann.

[RB97] Rosin C. and Belew R. (1997) New methods for competitive coevolution. *Evolutionary Computation* 15: 1–29.

[RGPC08] Ros F., Guillaume S., Pintore M., and Chretien J. R. (2008) Hybrid genetic algorithm for dual selection. *Pattern Analisys and Applications* 11: 179–198.

[SIL07] Saeys Y., Inza I., and Larrañaga P. (2007) A review of feature selection techniques in bioinformatics. *Bioinformatics* 23(19): 2507–2517.

[SLVH09] Sánchez A. M., Lozano M., Villar P., and Herrera F. (2009) Hybrid crossover operators with multiple descendents for real-coded genetic algorithms: Combining neighborhood-based crossover operators. *International Journal on Intelligent Systems* 24(5): 540–567.

[SWK09] Sun Y., Wong A. K. C., and Kamel M. S. (2009) Classification of imbalanced data: A review. *International Journal of Pattern Recognition and Artificial Intelligence* 23: 687–719.

[TDGH12] Triguero I., Derrac J., García S., and Herrera F. (2012) A taxonomy and experimental study on prototype generation for nearest neighbor classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews* 42(1): 86–100.

[TSK05] Tan P., Steinbach M., and Kumar V. (2005) *Introduction to Data Mining.* Addison-Wesley Longman Publishing Co., Inc., 1st edition.

[WAM97] Wettschereck D., Aha D. W., and Mohri T. (1997) A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. *Artificial Intelligence Review* 11: 273–314.

[WFH11] Witten I. H., Frank E., and Hall M. A. (2011) *Data Mining: Practical Machine Learning Tools and Techniques.* Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann.

[WK09] Wu X. and Kumar V. (Eds.) (2009) *The Top Ten Algorithms in Data Mining.* Data Mining and Knowledge Discovery. Chapman & Hall/CRC.

[WM97] Wilson D. and Martinez T. (1997) Improved heterogeneous distance functions. *Journal of Artificial Intelligence Research* 6: 1–34.

[YC98] Yang M. and Cheng C. (1998) On the edited fuzzy k-nearest neighbor rule. *IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics* 28(3): 461–466.

[YW06] Yang Q. and Wu X. (2006) 10 challenging problems in data mining research. *International Journal of Information Technology and Decision Making* 5(4): 597–604.

[Zad65] Zadeh L. (1965) Fuzzy sets. *Information and Control* 8(3): 338–353.

[Zad78] Zadeh L. (1978) Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems* 1: 3–28.

[ZLZ11] Zhai J.-H., Li N., and Zhai M.-Y. (2011) The condensed fuzzy k-nearest neighbor rule based on sample fuzzy entropy. In *Proceedings of the 2011 International Conference on Machine Learning and Cybernetics (ICMLC'11), Guilin, China, July 10-13*, pp. 282–286.